50272-101

REPORT DOCUMENTATION PAGE	1. REPORT NO. DOD/SW/MT-88/007a	2.	3. Recipient's Accession No.

DTIC FILE COPY

4. Title and Subtitle
Version Description Document for the Ada Compiler Validation Capability (Version 1.9): ANSI

5. Report Date

6.

7. Author(s)

8. Performing Organization Rept. No.

9. Performing Organization Name and Address
ASD/SCOL
Language Control Division
Area B, Bldg 676
Wright-Patterson AFB OH 45433-6503

10. Project/Task/Work Unit No.

11. Contract(C) or Grant(G) No.
(C)
(G)

13. Type of Report & Period Covered

14.

AD-A187 411

For magnetic tape, see:

16. Abstract (Limit: 200 words)

The Ada Compiler Validation Capability (ACVC) is a suite of tests designed to determine an Ada compiler's conformance to MIL-STD-1815A, the Ada Language Reference Manual (LRM). Version 1.9 was released in December 1986 and became the official ACVC version, replacing 1.8, in June 1987. ACVC version 1.9 will be the official version until June 1988 when it will be replace by ACVC version 1.10. Each of the 3122 tests in the ACVC 1.9 is designed to test a single test objective specified in the Ada Implementor's Guide (AIG), a document enumerating the LRM into discrete test objectives in order to aid compiler development.

17. Document Analysis a. Descriptors

The ACVC is a suite of tests designed to test an Ada Compiler's Conformance to MIL-STD-1815A,

b. Identifiers/Open-Ended Terms

c. COSATI Field/Group

18. Availability Statement	19. Security Class (This Report)	21. No. of Pages
		54
	20. Security Class (This Page)	22. Price

(See ANSI-Z39.18) *See Instructions on Reverse*

OPTIONAL FORM 272 (4-77)
(Formerly NTIS-35)
Department of Commerce

VERSION DESCRIPTION DOCUMENT
FOR THE
Ada® COMPILER VALIDATION CAPABILITY
(Version 1.9)

1 VERSION IDENTIFICATION

This document describes the Ada Compiler Validation Capability (ACVC), Version 1.9. This version consists of 3122 tests and 3591 test files. Also, there are 5 tests and 6 test files used to support the testing of the ACVC test suite. The differences between Version 1.8 and 1.9 are detailed in sections 2.0 through 4.0.

2 TEST MODIFICATIONS

In Version 1.9 of the ACVC, 431 tests were modified from the corresponding tests in Version 1.8. Modifications have been made to:

a) correct tests that were found to be incorrect in previous versions of the ACVC,
b) strengthen the tests by making the tests more comprehensive or incorporating more cases to check.

The term "former test" is used to refer to the test as it exists in Version 1.8 of the ACVC. The term "new test" is used to refer to the test as it exists in Version 1.9. The following tests have been modified from Version 1.8:

A26004A.ADA	AE2101B.ADA	AE2101D.ADA	B26005A.ADA
B33201B.ADA	B33203C.ADA	B36171A.ADA	B36201A.ADA
B37004A.ADA	B37101A.ADA	B37102A.ADA	B37201A.ADA
B37203A.ADA	B37309B.ADA	B37401A.ADA	B38003A.ADA
B38008A.ADA	B38101A.ADA	B43201D.ADA	B45116A.ADA
B45533A.ADA	B49006A.ADA	B4A010C.ADA	B62006C.ADA
B66001A.ADA	B74101B.ADA	B91001G.ADA	B95001B.ADA
B97103A.ADA	B99001A.ADA	BA1020C.ADA	BC1001A.ADA
BC1013A.ADA	BC1207A.ADA	BC3103A.ADA	BC3204C.DEP
BE2101E.ADA	BE3001A.ADA	C24113A.DEP	C24113B.DEP
C24113C.DEP	C24113D.DEP	C24113E.DEP	C24113F.DEP
C24113G.DEP	C24113H.DEP	C24113I.DEP	C24113J.DEP
C24113K.DEP	C24113L.DEP	C24113M.DEP	C24113N.DEP
C24113O.DEP	C24113P.DEP	C24113Q.DEP	C24113R.DEP
C24113S.DEP	C24113T.DEP	C24113U.DEP	C24113V.DEP
C24113W.DEP	C24113X.DEP	C24113Y.DEP	C32114A.ADA
C34001A.ADA	C34001C.ADA	C34001D.ADA	C34001F.ADA

®Ada is a registered trademark of the U.S. Government (Ada Joint Program Office).

C34002A.ADA	C34018A.ADA	C35508A.ADA	C35508B.ADA
C35703A.ADA	C35705A.DEP	C35705B.DEP	C35705C.DEP
C35705D.DEP	C35705E.DEP	C35705F.DEP	C35705G.DEP
C35705H.DEP	C35705I.DEP	C35705J.DEP	C35705K.DEP
C35705L.DEP	C35705M.DEP	C35705N.DEP	C35705O.DEP
C35705P.DEP	C35705Q.DEP	C35705R.DEP	C35705S.DEP
C35705T.DEP	C35705U.DEP	C35705V.DEP	C35705W.DEP
C35705X.DEP	C35705Y.DEP	C35707A.DEP	C35707B.DEP
C35707C.DEP	C35707D.DEP	C35707E.DEP	C35707F.DEP
C35707G.DEP	C35707H.DEP	C35707I.DEP	C35707J.DEP
C35707K.DEP	C35707L.DEP	C35707M.DEP	C35707N.DEP
C35707O.DEP	C35707P.DEP	C35707Q.DEP	C35707R.DEP
C35707S.DEP	C35707T.DEP	C35707U.DEP	C35707V.DEP
C35707W.DEP	C35707X.DEP	C35707Y.DEP	C35708A.DEP
C35708B.DEP	C35708C.DEP	C35708D.DEP	C35708E.DEP
C35708F.DEP	C35708G.DEP	C35708H.DEP	C35708I.DEP
C35708J.DEP	C35708K.DEP	C35708L.DEP	C35708M.DEP
C35708N.DEP	C35708O.DEP	C35708P.DEP	C35708Q.DEP
C35708R.DEP	C35708S.DEP	C35708T.DEP	C35708U.DEP
C35708V.DEP	C35708W.DEP	C35708X.DEP	C35708Y.DEP
C35802A.DEP	C35802B.DEP	C35802C.DEP	C35802D.DEP
C35802E.DEP	C35802F.DEP	C35802G.DEP	C35802H.DEP
C35802I.DEP	C35802J.DEP	C35802K.DEP	C35802L.DEP
C35802M.DEP	C35802N.DEP	C35802O.DEP	C35802P.DEP
C35802Q.DEP	C35802R.DEP	C35802S.DEP	C35802T.DEP
C35802U.DEP	C35802V.DEP	C35802W.DEP	C35802X.DEP
C35802Y.DEP	C35904A.ADA	C36202A.ADA	C36202B.ADA
C36301A.ADA	C36303A.ADA	C37008A.ADA	C37209A.ADA
C37213B.ADA	C41304A.ADA	C41404A.ADA	C43212A.ADA
C45241A.DEP	C45241B.DEP	C45241C.DEP	C45241D.DEP
C45241E.DEP	C45241F.DEP	C45241G.DEP	C45241H.DEP
C45241I.DEP	C45241J.DEP	C45241K.DEP	C45241L.DEP
C45241M.DEP	C45241N.DEP	C45241O.DEP	C45241P.DEP
C45241Q.DEP	C45241R.DEP	C45241S.DEP	C45241T.DEP
C45241U.DEP	C45241V.DEP	C45241W.DEP	C45241X.DEP
C45241Y.DEP	C45262A.ADA	C45262B.ADA	C45262C.ADA
C45321A.DEP	C45321B.DEP	C45321C.DEP	C45321D.DEP
C45321E.DEP	C45321F.DEP	C45321G.DEP	C45321H.DEP
C45321I.DEP	C45321J.DEP	C45321K.DEP	C45321L.DEP
C45321M.DEP	C45321N.DEP	C45321O.DEP	C45321P.DEP
C45321Q.DEP	C45321R.DEP	C45321S.DEP	C45321T.DEP
C45321U.DEP	C45321V.DEP	C45321W.DEP	C45321X.DEP
C45321Y.DEP	C45421A.DEP	C45421B.DEP	C45421C.DEP
C45421D.DEP	C45421E.DEP	C45421F.DEP	C45421G.DEP
C45421H.DEP	C45421I.DEP	C45421J.DEP	C45421K.DEP
C45421L.DEP	C45421M.DEP	C45421N.DEP	C45421O.DEP
C45421P.DEP	C45421Q.DEP	C45421R.DEP	C45421S.DEP
C45421T.DEP	C45421U.DEP	C45421V.DEP	C45421W.DEP
C45421X.DEP	C45421Y.DEP	C48008A.ADA	C4A011A.ADA
C4A013A.ADA	C52103X.ADA	C52104X.ADA	C54A03A.ADA
C58005B.ADA	C64103A.ADA	C87B50A.ADA	C87B62A.DEP
C87B62B.DEP	C87B62C.DEP	C92003A.ADA	C92005A.ADA

C93001A.ADA	C93003A.ADA	C94001A.ADA	C94002A.ADA
C94002B.ADA	C94006A.ADA	C95040D.ADA	C97113A.ADA
C97202A.ADA	CA1012A.DEP	CA2009C.DEP	CA2009F.DEP
CE2102C.TST	CE2102G.ADA	CE2104B.ADA	CE2104C.ADA
CE2104D.ADA	CE2105A.ADA	CE2106A.ADA	CE2107A.ADA
CE2107B.ADA	CE2107C.ADA	CE2107D.ADA	CE2107E.ADA
CE2107F.ADA	CE2108A.ADA	CE2108B.ADA	CE2108C.ADA
CE2108D.ADA	CE2109A.ADA	CE2111D.ADA	CE2111G.ADA
CE2111H.ADA	CE2201A.ADA	CE2201B.ADA	CE2201C.ADA
CE2201F.ADA	CE2204A.ADA	CE2204B.ADA	CE2210A.ADA
CE2401A.ADA	CE2401B.ADA	CE2401C.ADA	CE2401E.ADA
CE2401F.ADA	CE2404A.ADA	CE2406A.ADA	CE2407A.ADA
CE2409A.ADA	CE3103A.ADA	CE3104A.ADA	CE3109A.ADA
CE3110A.ADA	CE3111A.ADA	CE3111B.ADA	CE3111C.ADA
CE3111D.ADA	CE3111E.ADA	CE3112A.ADA	CE3114B.ADA
CE3115A.ADA	CE3203A.ADA	CE3208A.ADA	CE3301A.ADA
CE3301B.ADA	CE3301C.ADA	CE3302A.ADA	CE3402A.ADA
CE3402B.ADA	CE3402C.ADA	CE3402D.ADA	CE3403A.ADA
CE3403B.ADA	CE3403C.ADA	CE3403E.ADA	CE3403F.ADA
CE3404A.ADA	CE3404B.ADA	CE3404C.ADA	CE3405A.ADA
CE3405B.ADA	CE3405C.ADA	CE3405D.ADA	CE3406A.ADA
CE3406B.ADA	CE3406C.ADA	CE3406D.ADA	CE3407A.ADA
CE3407B.ADA	CE3407C.ADA	CE3408A.ADA	CE3408B.ADA
CE3408C.ADA	CE3409A.ADA	CE3409C.ADA	CE3409D.ADA
CE3409E.ADA	CE3409F.ADA	CE3410A.ADA	CE3410C.ADA
CE3410D.ADA	CE3410E.ADA	CE3410F.ADA	CE3411A.ADA
CE3412A.ADA	CE3413A.ADA	CE3413C.ADA	CE3602A.ADA
CE3602B.ADA	CE3602C.ADA	CE3602D.ADA	CE3603A.ADA
CE3604A.ADA	CE3605A.ADA	CE3605B.ADA	CE3605C.ADA
CE3605D.ADA	CE3605E.ADA	CE3606A.ADA	CE3606B.ADA
CE3704A.ADA	CE3704B.ADA	CE3704D.ADA	CE3704E.ADA
CE3704F.ADA	CE3704M.ADA	CE3704N.ADA	CE3704O.ADA
CE3706D.ADA	CE3706F.ADA	CE3804A.ADA	CE3804B.ADA
CE3804C.ADA	CE3804D.ADA	CE3804E.ADA	CE3804G.ADA
CE3804I.ADA	CE3804K.ADA	CE3804M.ADA	CE3805A.ADA
CE3805B.ADA	CE3806A.ADA	CE3806D.ADA	CE3806E.ADA
CE3905A.ADA	CE3905B.ADA	CE3905C.ADA	CE3905L.ADA
CE3906A.ADA	CE3906B.ADA	CE3906C.ADA	CE3906E.ADA
CE3906F.ADA	D4A002B.ADA	E24101A.TST	

Please note that three tests, A26004A, C34001D, and C34001F, were modified and the file extension has changed.

In addition to the test modifications noted above, all tests were modified to remove the language version suffix (-AB or -B) from the official test name. The language version suffix will no longer be a part of the official ACVC test name.

3 TEST DELETIONS

The following 97 tests appear in ACVC Version 1.8, but not in ACVC Version 1.9. Several of the deleted tests were renamed to new test names in ACVC 1.9 to correspond to the naming conventions provided by the Implementer's Guide.

A34008B.ADA	C32107B.ADA	B34008A.ADA	B35301A.ADA
B35A03A.ADA	B45402A.ADA	B4A006A.ADA	B920BDA.ADA
B950ACA.ADA	B950BAA.ADA	B950DHA.ADA	BC10ADA.ADA
BC10AEB.ADA	BC10AEC.ADA	BC11ABA.ADA	BC11ACA.ADA
BC12ABA.ADA	BC13ABA.ADA	BC30ABA.ADA	BC30ACA.ADA
BC31ABA.ADA	BC31ACA.ADA	BC31ADA.ADA	BC32ABA.ADA
BC32ADA.ADA	BC33ABA.ADA	BC33ACA.ADA	BC33ADA.ADA
BC33AEA.ADA	C34001B.ADA	C34001E.DEP	C34001G.DEP
C34001H.ADA	C34001I.ADA	C34001K.ADA	C34001L.ADA
C34001M.ADA	C34001N.ADA	C34001O.ADA	C34001P.ADA
C34001Q.ADA	C34001R.ADA	C34001T.ADA	C34002B.ADA
C35104A.ADA	C38007A.ADA	C45401A.ADA	C45401B.ADA
C45424A.DEP	C45424B.DEP	C45424C.DEP	C45424D.DEP
C45424E.DEP	C45424F.DEP	C45424G.DEP	C45424H.DEP
C45424I.DEP	C45424J.DEP	C45424K.DEP	C45424L.DEP
C45424M.DEP	C45424N.DEP	C45424O.DEP	C45424P.DEP
C45424Q.DEP	C45424R.DEP	C45424S.DEP	C45424T.DEP
C45424U.DEP	C45424V.DEP	C45424W.DEP	C45424X.DEP
C45424Y.DEP	C45526A.ADA	C4A001A.ADA	C4A003A.ADA
C910AHA.ADA	C930ABA.ADA	C930AFA.ADA	C930AJA.ADA
C940ABA.ADA	C940ACA.ADA	C940ACB.ADA	C940ADA.ADA
C940AHA.ADA	C940BAA.ADA	C940BBA.ADA	CA3005A.ADA
CA3005B.ADA	CA3005C.ADA	CA3005D.ADA	CE2201D.DEP
CE2201E.DEP	CE2401D.DEP	E94004A.ADA	E94004B.ADA
E94004C.ADA			

In addition, the support package VAR_STRINGS and the four tests (CZ1201A .. D) that check the VAR_STRINGS package have been deleted.

4 NEW TESTS

The following 820 tests have been added to Version 1.9 of the ACVC Test Suite. A test is considered "new" if the test name did not exist in ACVC Version 1.8. Several of the new tests were renamed from former tests in ACVC Version 1.8 to correspond to the naming conventions provided by the Implementer's Guide.

A22006C.ADA	A22006D.ADA	A22006E.ADA	A22006F.ADA
A26007A.TST	A27003A.ADA	A27004A.ADA	A28002C.ADA
A28004A.ADA	A29003A.ADA	A33003A.ADA	A34017C.ADA

A35101B.ADA A35402A.ADA A35502Q.ADA A35502R.ADA
A35801A.ADA A35801B.ADA A35801E.ADA A35801F.ADA
A35902C.ADA A37312A.ADA A39005B.DEP A39005C.DEP
A39005D.DEP A39005E.DEP A39005F.DEP A39005G.DEP
A41307A.ADA A41307C.ADA A87B59A.ADA A91001I.ADA
A95001C.ADA AA2010A.ADA AA2012A.ADA AA5008X.ADA
AD1A01A.ADA AD1A01B.ADA AD1A01C.ADA AD1A01D.ADA
AE2101G.ADA AE2101I.ADA AE2113A.ADA AE2113B.ADA
AE3002G.ADA B22005A.ADA B22005B.ADA B22005C.ADA
B22005D.ADA B22005E.ADA B22005F.ADA B22005G.ADA
B22005H.ADA B22005I.ADA B22005J.ADA B22005K.ADA
B22005L.ADA B22005M.ADA B22005N.ADA B22005O.ADA
B22005P.ADA B22005Q.ADA B22005R.ADA B22005S.ADA
B22005T.ADA B22005U.ADA B22005V.ADA B22005W.ADA
B22005X.ADA B22005Y.ADA B22005Z.ADA B24007A.ADA
B24009A.ADA B24104A.ADA B24205A.ADA B24206A.ADA
B24206B.ADA B25002A.ADA B25002B.ADA B25004B.ADA
B26001A.ADA B27005A.ADA B28001A.ADA B28001B.ADA
B28001C.ADA B28001D.ADA B28001E.ADA B28001F.ADA
B28001G.ADA B28001H.ADA B28001I.ADA B28001J.ADA
B28001K.ADA B28001L.ADA B28001M.ADA B28001N.ADA
B28001O.ADA B28001P.ADA B28001Q.ADA B28001R.ADA
B28001S.ADA B28001T.ADA B28001U.ADA B28001V.ADA
B28001W.ADA B28003A.ADA B28003C.ADA B2A004A.ADA
B2A005A.ADA B2A005B.ADA B2A007A.ADA B2A010A.ADA
B32101A.ADA B33302A.ADA B34001B.ADA B34001E.ADA
B34002B.ADA B34003B.ADA B34004B.ADA B34007B.ADA
B34007E.ADA B34007H.ADA B34007K.ADA B34007N.ADA
B34007Q.ADA B34007T.ADA B34008B.ADA B34009B.ADA
B34009E.ADA B34009H.ADA B34009K.ADA B34017A.ADA
B34017B.ADA B35004A.ADA B35302A.ADA B35401A.ADA
B35401B.ADA B35403A.ADA B35501B.ADA B35506C.ADA
B35506D.ADA B35801C.ADA B35803A.ADA B35901C.ADA
B35901D.ADA B35A01A.ADA B36001A.ADA B36002A.ADA
B36307A.ADA B37104A.ADA B37106A.ADA B37212A.ADA
B37312B.ADA B38003B.ADA B38009A.ADA B38009B.ADA
B38101C.ADA B38101D.ADA B38103D.ADA B38103E.ADA
B38203A.ADA B38204A.ADA B39004A.ADA B39004B.ADA
B39004C.ADA B39004D.ADA B39004E.ADA B39004F.ADA
B39004G.ADA B39004H.ADA B39004I.ADA B39004J.ADA
B39004K.ADA B39004L.ADA B41305A.ADA B41324B.ADA
B41325B.ADA B41327B.ADA B44004A.ADA B44004B.ADA
B44004C.ADA B44004D.ADA B44004E.ADA B45301A.ADA
B45301B.ADA B45301C.ADA B45302A.ADA B45341A.ADA
B45535A.ADA B45537A.ADA B45601A.ADA B45661A.ADA
B46002A.ADA B46004A.ADA B46004B.ADA B46004C.ADA
B46004D.ADA B46004E.ADA B47001A.ADA B49002A.ADA
B49003A.ADA B49004A.ADA B49005A.ADA B49007A.ADA
B49008A.ADA B49008B.ADA B49008C.ADA B49009A.ADA
B49009B.ADA B49009C.ADA B49010A.ADA B49011A.ADA
B51003B.ADA B74404A.ADA B91001H.ADA B91003D.ADA
B92001A.ADA B92001B.ADA B95069C.ADA B97103F.ADA

B97103G.ADA	BA1109A.ADA	BA1110A.ADA	BB1006A.ADA
BB1006B.ADA	BC1005A.ADA	BC1014B.ADA	BC1109A.ADA
BC1109B.ADA	BC1109C.ADA	BC1109D.ADA	BC1110A.ADA
BC1201F.ADA	BC1201G.ADA	BC1201H.ADA	BC1201I.ADA
BC1201J.ADA	BC1201K.ADA	BC1201L.ADA	BC1206A.ADA
BC1230A.ADA	BC1303F.ADA	BC2001D.ADA	BC2001E.ADA
BC2004B.ADA	BC3005B.ADA	BC3104A.ADA	BC3105A.ADA
BC3604A.ADA	BC3604B.ADA	BE2101J.ADA	C24207A.ADA
C25001A.ADA	C25001B.ADA	C25003A.ADA	C25004A.ADA
C2A006A.ADA	C2A008A.ADA	C2A009A.TST	C32001A.ADA
C32001B.ADA	C32001C.ADA	C32001D.ADA	C32001E.ADA
C32107A.ADA	C32107C.ADA	C32108A.ADA	C32108B.ADA
C32111A.ADA	C32112A.ADA	C32112B.ADA	C32113A.ADA
C32115A.ADA	C32117A.ADA	C34002C.ADA	C34003A.ADA
C34003C.ADA	C34004A.ADA	C34007A.ADA	C34007D.ADA
C34007F.ADA	C34007G.ADA	C34007I.ADA	C34007M.ADA
C34007P.ADA	C34007R.ADA	C34007S.ADA	C34007U.ADA
C34012A.ADA	C34015B.ADA	C34016B.ADA	C35102A.ADA
C35106A.ADA	C35502A.ADA	C35502B.ADA	C35502C.ADA
C35502D.TST	C35502E.ADA	C35502F.TST	C35502G.ADA
C35502H.ADA	C35502I.DEP	C35502J.DEP	C35502K.ADA
C35502L.ADA	C35502M.DEP	C35502N.DEP	C35502O.ADA
C35502P.ADA	C35503A.ADA	C35503B.ADA	C35503C.ADA
C35503D.TST	C35503E.ADA	C35503F.TST	C35503G.ADA
C35503H.ADA	C35503K.ADA	C35503L.ADA	C35503O.ADA
C35503P.ADA	C35505C.ADA	C35505D.ADA	C35507A.ADA
C35507B.ADA	C35507C.ADA	C35507E.ADA	C35507G.ADA
C35507H.ADA	C35507I.ADA	C35507J.DEP	C35507K.ADA
C35507L.ADA	C35507M.DEP	C35507N.DEP	C35507O.ADA
C35507P.ADA	C35508C.ADA	C35508E.ADA	C35508G.ADA
C35508H.ADA	C35508I.DEP	C35508J.DEP	C35508K.ADA
C35508L.ADA	C35508M.DEP	C35508N.DEP	C35508O.ADA
C35508P.ADA	C35710A.ADA	C35711B.ADA	C35712A.ADA
C35712B.ADA	C35712C.ADA	C35801D.ADA	C35802Z.DEP
C35902A.ADA	C35902B.ADA	C35902D.ADA	C35903A.ADA
C35904B.ADA	C35A02A.ADA	C35A03A.ADA	C35A03B.ADA
C35A03C.ADA	C35A03D.ADA	C35A03E.ADA	C35A03N.ADA
C35A03O.ADA	C35A03P.ADA	C35A03Q.ADA	C35A03R.ADA
C35A04A.ADA	C35A04B.ADA	C35A04C.ADA	C35A04D.ADA
C35A04N.ADA	C35A04O.ADA	C35A04P.ADA	C35A04Q.ADA
C35A05A.ADA	C35A05D.ADA	C35A05N.ADA	C35A05Q.ADA
C35A06A.ADA	C35A06B.ADA	C35A06D.ADA	C35A06N.ADA
C35A07A.ADA	C35A07B.ADA	C35A07C.ADA	C35A07D.ADA
C35A07N.ADA	C35A07O.ADA	C35A07P.ADA	C36003A.ADA
C36004A.ADA	C36104A.ADA	C36104B.ADA	C36202C.ADA
C36203A.ADA	C36204B.ADA	C36204C.ADA	C37002A.ADA
C37006A.ADA	C37010A.ADA	C37010B.ADA	C37102B.ADA
C37107A.ADA	C37108B.ADA	C37206A.ADA	C37207A.ADA
C37209B.ADA	C37210A.ADA	C37211A.ADA	C37211B.ADA
C37211C.ADA	C37211D.ADA	C37211E.ADA	C37213A.ADA
C37213C.ADA	C37213D.ADA	C37213E.ADA	C37213F.ADA
C37213G.ADA	C37213H.ADA	C37213J.ADA	C37215A.ADA

C37215B.ADA C37215C.ADA C37215D.ADA C37215E.ADA
C37215F.ADA C37215G.ADA C37215H.ADA C37402A.ADA
C37403A.ADA C37405A.ADA C38002A.ADA C38002B.ADA
C38004B.ADA C38102D.ADA C38102E.ADA C38108A.ADA
C38108B.ADA C38108C.ADA C38108D.ADA C38201A.ADA
C38202A.ADA C39005A.ADA C39006A.ADA C39006B.ADA
C39006C.ADA C39006D.ADA C39006E.ADA C39006F.ADA
C39007A.ADA C41104A.ADA C41304B.ADA C41307D.ADA
C41308C.ADA C41308D.ADA C41320A.ADA C41321A.ADA
C41322A.ADA C41323A.ADA C41324A.ADA C41325A.ADA
C41326A.ADA C41327A.ADA C41328A.ADA C41401A.ADA
C41402A.ADA C41403A.ADA C42007A.ADA C42007B.ADA
C42007C.ADA C42007D.ADA C42007E.ADA C42007F.ADA
C42007G.ADA C42007H.ADA C42007I.ADA C42007J.ADA
C42007K.ADA C43003A.ADA C45111A.ADA C45111B.ADA
C45111C.ADA C45111D.ADA C45111E.ADA C45112A.ADA
C45112B.ADA C45113A.ADA C45114B.ADA C45202B.ADA
C45211A.ADA C45231A.ADA C45231B.DEP C45231C.DEP
C45231D.TST C45232A.TST C45232B.ADA C45242B.ADA
C45251A.ADA C45252A.ADA C45252B.ADA C45253A.ADA
C45264C.ADA C45265A.ADA C45271A.ADA C45272A.ADA
C45273A.ADA C45281A.ADA C45282A.ADA C45282B.ADA
C45304A.ADA C45304B.DEP C45304C.DEP C45331A.ADA
C45331D.ADA C45332A.ADA C45344A.ADA C45431A.ADA
C45502A.ADA C45502B.DEP C45502C.DEP C45503A.ADA
C45503B.DEP C45503C.DEP C45504A.ADA C45504B.DEP
C45504C.DEP C45504D.ADA C45504E.DEP C45504F.DEP
C45524A.DEP C45524B.DEP C45524C.DEP C45524D.DEP
C45524E.DEP C45524F.DEP C45524G.DEP C45524H.DEP
C45524I.DEP C45524J.DEP C45524K.DEP C45524L.DEP
C45524M.DEP C45524N.DEP C45524O.DEP C45524P.DEP
C45524Q.DEP C45524R.DEP C45524S.DEP C45524T.DEP
C45524U.DEP C45524V.DEP C45524W.DEP C45524X.DEP
C45524Y.DEP C45524Z.DEP C45531A.DEP C45531B.DEP
C45531C.DEP C45531D.DEP C45531E.DEP C45531F.DEP
C45531G.DEP C45531H.DEP C45531I.DEP C45531J.DEP
C45531K.DEP C45531L.DEP C45531M.DEP C45531N.DEP
C45531O.DEP C45531P.DEP C45532A.DEP C45532B.DEP
C45532C.DEP C45532D.DEP C45532E.DEP C45532F.DEP
C45532G.DEP C45532H.DEP C45532I.DEP C45532J.DEP
C45532K.DEP C45532L.DEP C45532M.DEP C45532N.DEP
C45532O.DEP C45532P.DEP C45611A.ADA C45611B.DEP
C45611C.DEP C45613A.ADA C45613B.DEP C45613C.DEP
C45614A.ADA C45614B.DEP C45614C.DEP C45631A.ADA
C45631B.DEP C45631C.DEP C45632A.ADA C45632B.DEP
C45632C.DEP C45641A.DEP C45641B.DEP C45641C.DEP
C45641D.DEP C45641E.DEP C45641F.DEP C45641G.DEP
C45641H.DEP C45641I.DEP C45641J.DEP C45641K.DEP
C45641L.DEP C45641M.DEP C45641N.DEP C45641O.DEP
C45641P.DEP C45641Q.DEP C45641R.DEP C45641S.DEP
C45641T.DEP C45641U.DEP C45641V.DEP C45641W.DEP
C45641X.DEP C45641Y.DEP C45651A.ADA C45662A.ADA

C45662B.ADA	C46011A.ADA	C46012A.DEP	C46012B.DEP
C46012C.DEP	C46012D.DEP	C46012E.DEP	C46012F.DEP
C46012G.DEP	C46012H.DEP	C46012I.DEP	C46012J.DEP
C46012K.DEP	C46012L.DEP	C46012M.DEP	C46012N.DEP
C46012O.DEP	C46012P.DEP	C46012Q.DEP	C46012R.DEP
C46012S.DEP	C46012T.DEP	C46012U.DEP	C46012V.DEP
C46012W.DEP	C46012X.DEP	C46012Y.DEP	C46012Z.DEP
C46014A.ADA	C46041A.ADA	C46042A.ADA	C46043A.ADA
C46043B.ADA	C46044A.ADA	C46044B.ADA	C46051A.ADA
C46052A.ADA	C46053A.ADA	C46054A.ADA	C47002A.ADA
C47002B.ADA	C47002C.ADA	C47002D.ADA	C47003A.ADA
C47004A.ADA	C47005A.ADA	C47006A.ADA	C47007A.ADA
C47008A.ADA	C47009A.ADA	C47009B.ADA	C49020A.ADA
C49021A.ADA	C49022A.ADA	C49022B.ADA	C49022C.ADA
C49023A.ADA	C49024A.ADA	C49025A.ADA	C49026A.ADA
C4A006A.ADA	C4A010B.ADA	C4A010D.ADA	C4A012B.ADA
C4A013B.ADA	C64109H.ADA	C64109I.ADA	C64109J.ADA
C64109K.ADA	C64109L.ADA	C85018A.ADA	C85018B.ADA
C85019A.ADA	C87B62D.DEP	C91006A.ADA	C93004D.ADA
C93004F.ADA	C93005B.ADA	C93008A.ADA	C93008B.ADA
C94001B.ADA	C94001C.ADA	C94002D.ADA	C94002E.ADA
C94002F.ADA	C94002G.ADA	C94004A.ADA	C94004B.ADA
C94004C.ADA	C94008A.ADA	C94008B.ADA	C94008C.ADA
C94008D.ADA	C94010A.ADA	C94011A.ADA	C95085B.ADA
C95085C.ADA	C95085D.ADA	C95085E.ADA	C95085F.ADA
C95085G.ADA	C95085H.ADA	C95085I.ADA	C95085J.ADA
C95085K.ADA	C95085L.ADA	C95085M.ADA	C95085N.ADA
C95085O.ADA	C95086B.ADA	C95086C.ADA	C95086D.ADA
C95086E.ADA	C95086F.ADA	C95087B.ADA	C95087C.ADA
C95087D.ADA	C97112A.ADA	C97116A.ADA	C97117A.ADA
C97117B.ADA	C97117C.ADA	C97118A.ADA	C97120A.ADA
C97120B.ADA	C97201B.ADA	C97201C.ADA	C97203C.ADA
C97204B.ADA	C97205A.ADA	C97205B.ADA	C97301A.ADA
C97301B.ADA	C97301C.ADA	C97301D.ADA	C97301E.ADA
C97302A.ADA	C97303C.ADA	C97304B.ADA	C97305A.ADA
C97305B.ADA	C97305C.ADA	C97305D.ADA	C97307A.ADA
C99004A.ADA	C9A008A.ADA	CA3011A.DEP	CA5006A.ADA
CB1005A.ADA	CB3003B.ADA	CC1018A.ADA	CC1221A.ADA
CC1222A.ADA	CC1224A.ADA	CC1311A.ADA	CC1311B.ADA
CC3015A.ADA	CC3121A.ADA	CC3601A.ADA	CC3603A.ADA
CE2102H.TST	CE2102K.ADA	CE2105B.ADA	CE2106B.ADA
CE2107G.ADA	CE2107H.ADA	CE2107I.ADA	CE2109B.ADA
CE2109C.ADA	CE2115A.ADA	CE2115B.ADA	CE2201G.ADA
CE2208B.ADA	CE2401H.ADA	CE2411A.ADA	E28002A.ADA
E28002B.ADA	E28002D.ADA	E28005C.ADA	E28005D.ADA
EE2201D.ADA	EE2201E.ADA	EE2401D.ADA	EE2401G.ADA

ACVC 1.9 Tape Information

1 INTRODUCTION

The ACVC tests are written in ANSI Standard format on one tape volume. There are 3598 test files in the ACVC Version 1.9. On the ANSI Standard format tape, there are a total of 143 files. Each of 31 of these files contains multiple tests packed onto a single file to save space on the tape volume. There are 110 single test files for tests containing non-graphical characters or long text lines, one Ada®program to retrieve individual tests from the packed multi-test files, and a list of all the ACVC Version 1.9 test files. The tape is written at a density of 1600 bpi.

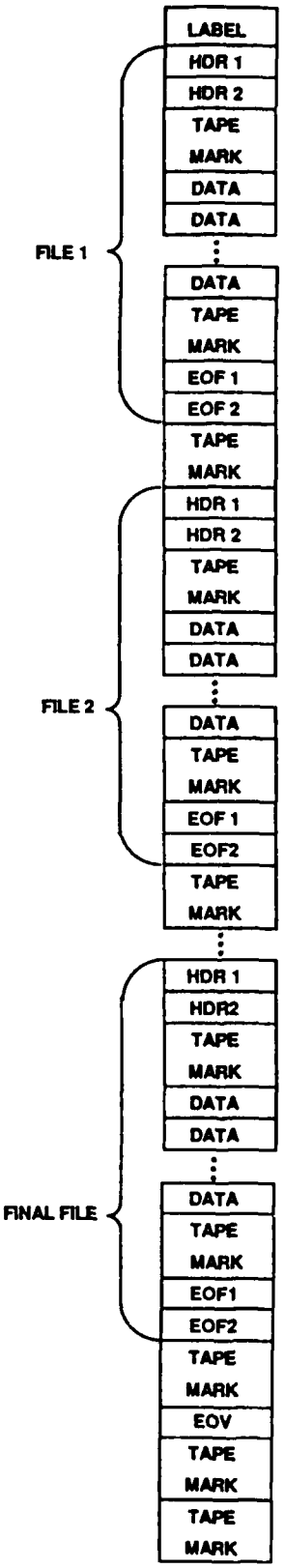
2 TAPE FORMAT

The ACVC tests are written on one tape volume. Each file on the tape is written as a set of variable-length records in blocks of 2048 characters, using the ANSI variable-length record (D), blocked format. Each file in this format has the structure shown in Figure 1.

Each data block is 2048 characters in length. Each record within a block represents a line of text. The first four characters of each record make up the Record Control Word (RCW). The record length (the number of characters contained in the record) is expressed as a sequence of four ASCII characters occupying the entire RCW and yielding a decimal value. The record length includes the length of the RCW. Hence, a blank (empty) line is represented by the record 0004. A line containing the characters ADA would be represented as 0007ADA. Each record is completely contained within a block. If a record including the four-character RCW cannot be contained in the space remaining in a block, then each remaining byte in the block is set to a circumflex (^), and the record is placed in the next block. Figure 2 shows the format of a file that partially fills a data block.

®Ada is a registered trademark of the U.S. Government Ada Joint Program Office (AJPO).

ACVC DISTRIBUTION TAPE FORMAT



A-D8774

Figure 1.

```
0018-- C35702B.DEP00040050-- CHECK THAT LONG_FLOAT'DIGITS > FLOA
T'DIGITS00040020-- BAW 5 SEPT 800018-- SPS 2/18/8300040016WITH R
EPORT;0024PROCEDURE C35702B IS00040020     USE REPORT;00040075BE
GIN TEST("C35702B","CHECK IF LONG_FLOAT HAS MORE DIGITS THAN FLO
AT");00040046      IF FLOAT'DIGITS >= LONG_FLOAT'DIGITS0070
 THEN FAILED("LONG_FLOAT HAS NOT MORE PRECISION THAN FLOAT");001
7      END IF;00040017      RESULT;00040016END C35702B;^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

Figure 2.

3 ACVC TEST FILES

The ACVC tests are stored on the tape using a method that packs many tests into one file. There are 31 multi-test-formatted files on the tape (named ACVC1...ACVC31), 32 separate files containing individual tests with special control characters, and 78 separate files containing individual tests with lines over 72 characters long. Files ACVC1...ACVC31 do not contain any text lines over 72 characters long. Also on the tape is an Ada program UNPACK that can be used on some systems to retrieve the individual ACVC tests from the packed files, as well as one file that contains the names of all the ACVC tests included in the files, ACVC9.LST. This is the order in which the files appear on the tape:

- File 1 : UNPACK.ADA
- File 2 : ACVC9.LST
- Files 3 - 33 : ACVC1...ACVC31
- Files 34 - 65 : Individual files with nongraphic ASCII characters
- Files 66 - 143 : Individual files with long lines

Figure 3 shows how the files are structured on the tape. The multifile test files have this repeating structure:

- <<< test name
- an ACVC test
- >>> test name (blank) number of lines in the test

Figure 4 shows an example of the structure of the ACVC multifile format. The number of lines given at the end of each file following the >>> can be used to check that the file was unpacked properly.

UNPACK will work to separate the individual tests if the packed file read from the tape is in a format that conforms to the format used by the implementation of the Ada package TEXT_IO. UNPACK reads a packed file of tests and puts each of the tests into a single file. UNPACK requires as input the name of the file to be unpacked. New files are created using the ACVC test name as the file name. UNPACK will need to be run once for each packed file.

The program text may have to be modified to coincide with the file-naming conventions of the system onto which the tests will be loaded. The TEXT_IO.CREATE command on line 36 of the program UNPACK creates a file using the returned value from the function TESTFILE_NAME. Modifications may be made inside this function to provide conformance to the file-naming conventions of the system. Without modification, program UNPACK will try to name the new files it creates the names in the packed files that are listed after "<<< ". In figure 4, C35702B.DEP would be the name of the new file created which would not agree with the file naming conventions of some systems.

MULTIFILE FORMAT

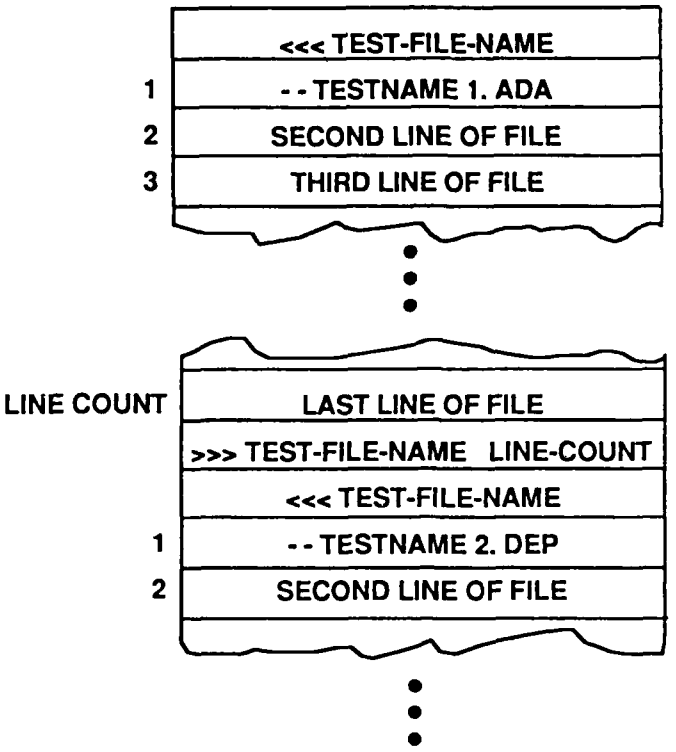


Figure 3.

```
<<< C35702B.DEP
- - C35702B.DEP

- - CHECK THAT LONG_FLOAT'DIGITS > FLOAT'DIGITS

- - BAW 5 SEPT 80
- - SPS 2/18/83

WITH REPORT;
PROCEDURE C35702B IS

    USE REPORT;

BEGIN TEST("C35702B","CHECK IF LONG_FLOAT HAS MORE DIGITS THAN FLOAT");

    IF FLOAT'DIGITS >= LONG_FLOAT'DIGITS
    THEN FAILED("LONG_FLOAT HAS NOT MORE PRECISION THAN FLOAT");
    END IF;

RESULT;

END C35702B;
>>> C35702B.DEP 21
```

Figure 4.

The specification, REPSPEC, and body, REPBODY, of package REPORT are included in the packed file ACVC31. Package REPORT is used in many of the ACVC tests. Therefore, ACVC31 should be unpacked and REPSPEC and REPBODY compiled before any of the ACVC tests are run.

4 MULTIFILE TEST FILE CONTENTS

The following list shows each multifile test file with its corresponding number of individual files in it. This list may be helpful in unpacking the files. UNPACK will display the number of test files it retrieves from each packed file.

FILE	FILES CREATED
ACVC1	193
ACVC2	119
ACVC3	149
ACVC4	206
ACVC5	99
ACVC6	382
ACVC7	133
ACVC8	99
ACVC9	124
ACVC10	77
ACVC11	35
ACVC12	56
ACVC13	90
ACVC14	99
ACVC15	61
ACVC16	62
ACVC17	42
ACVC18	68
ACVC19	51
ACVC20	75
ACVC21	77
ACVC22	92
ACVC23	98
ACVC24	118
ACVC25	87
ACVC26	251
ACVC27	100
ACVC28	96
ACVC29	80
ACVC30	241
ACVC31	28

5 FILES WITH NONGRAPHIC CHARACTERS

32 of the files in the ACVC test suite contain ASCII control characters. These files are stored individually on the distribution tape. The following list shows the files with the corresponding control characters in the file.

A22006C.ADA	CR, VT, LF, FF
A22006E.ADA	CR, VT, LF, FF
B22005A.ADA	SOH
B22005B.ADA	STX
B22005C.ADA	ETX
B22005D.ADA	EOT
B22005E.ADA	ENQ
B22005F.ADA	ACK
B22005G.ADA	BEL
B22005H.ADA	BS
B22005I.ADA	HT
B22005J.ADA	LF
B22005K.ADA	VT
B22005L.ADA	FF
B22005M.ADA	CR
B22005N.ADA	SO
B22005O.ADA	SI
B22005P.ADA	DLE
B22005Q.ADA	DC1
B22005R.ADA	DC2
B22005S.ADA	DC3
B22005T.ADA	DC4
B22005U.ADA	NAK
B22005V.ADA	SYN
B22005W.ADA	ETB
B22005X.ADA	CAN
B22005Y.ADA	EM
B22005Z.ADA	SUB
B25002A.ADA	Control Characters SOH through BS, and SO through SUB, ESC, FS, GS, RS, US, NUL, DEL
B25002B.ADA	HT, VT, CR, LF, FF
B26005A.ADA	Control Characters SOH through SUB, ESC, FS, GS, RS, US, DEL
B27005A.ADA	Control Characters SOH through BS, and SO through SUB, NUL, ESC, FS, GS, RS, US, DEL

6 FILES WITH LONG TEXT LINES

78 of the files in the ACVC test suite contain lines that are longer than 72 characters. These files are stored individually on the distribution tape. The following table shows the files that have long lines along with the length of the longest line in the file.

TEST NAMES	LENGTH (characters)	TEST NAMES	LENGTH (characters)
B23003F.TST	74	C35508H.ADA	77
B49006A.ADA	73	C35508J.DEP	73
B49009A.ADA	75	C35508O.ADA	78
BC3604A.ADA	73	C35508P.ADA	77
C24113C.DEP	78	C35802Y.DEP	73
C24113D.DEP	86	C35802Z.DEP	75
C24113E.DEP	95	C37206A.ADA	76
C24113F.DEP	106	C45242B.ADA	73
C24113G.DEP	114	C45252B.ADA	73
C24113H.DEP	123	C45532B.DEP	73
C24113I.DEP	134	C45532F.DEP	73
C24113J.DEP	142	C45532J.DEP	73
C24113K.DEP	151	C45532N.DEP	73
C24113L.DEP	162	C45621G.DEP	73
C24113M.DEP	170	C45621H.DEP	73
C24113N.DEP	179	C45621I.DEP	74
C24113O.DEP	190	C45621J.DEP	75
C24113P.DEP	198	C45621K.DEP	76
C24113Q.DEP	207	C45621L.DEP	77
C24113R.DEP	218	C45621M.DEP	78
C24113S.DEP	226	C45621N.DEP	78
C24113T.DEP	235	C45621O.DEP	79
C24113U.DEP	247	C45621P.DEP	80
C24113V.DEP	255	C45621Q.DEP	81
C24113W.DEP	264	C45621R.DEP	82
C24113X.DEP	275	C45621S.DEP	83
C24113Y.DEP	283	C45621T.DEP	83
C35502P.ADA	79	C45621U.DEP	84
C35503D.TST	74	C45621V.DEP	85
C35503E.ADA	77	C45621W.DEP	86
C35503F.TST	78	C45621X.DEP	87
C35503G.ADA	74	C45621Y.DEP	88
C35503H.ADA	76	C45621Z.DEP	89
C35503K.ADA	78	C52104L.ADA	73
C35503L.ADA	73	C54A24B.ADA	73
C35503O.ADA	74	C87B06A.ADA	73
C35503P.ADA	77	C87B19A.ADA	75
C35508E.ADA	77	C97301B.ADA	78
C35508G.ADA	78	MACRO.DFS	180

7 ACVC TEST FILE NAMES

ACVC test file names may not conform to the naming conventions of the computer system on which the files are to be read. The test file names follow certain conventions. However, certain portions of a name provide information but may be removed in order to shorten a name while preserving uniqueness.

The maximum length of a test file descriptor is 13 characters that consists of a file name and file type. The last four characters designate the file type--a period followed by three letters--and can be deleted without creating duplicate file names. When a file name consists of nine characters followed by a period, the ninth character is always M and can be deleted. All other test file names are seven or eight characters in length before a period occurs. Finally, the first letter of each test file name can be safely deleted without creating duplicate file names. If all these deletions are performed, test files will have names that are six to nine characters in length.

8 TEST MODIFICATION

The Ada tests will be periodically updated and corrected. Their correctness is not guaranteed either by the ACVC Maintenance Organization (AMO) or by its contractor. Please report errors in writing to

Mrs. Georgeanne Chitwood
ACVC Maintenance Organization
ASD/SCOL
Wright-Patterson AFB, OH 45433-6503

You will receive notification of changes to this version of the ACVC on a periodic basis. Please notify the AMO of any change of address.

ACVC test names have the following form: NAME.TYPE. The NAME component has the following interpretation, where the first column below indicates the character position in NAME and the second column, the meaning of that position:

1	Class of test (A, B, C, D, E, L).
2	Implementers' Guide chapter number (in hexadecimal).
3	Implementers' Guide section number within a chapter (in hexadecimal).
4	Implementers' Guide subsection number or letter.
5, 6	Implementers' Guide test objective number (two digit decimal number). There are currently 4 tests (listed in Appendix F) which have letters in this position of the test name. These test names will be renamed to correspond to the Implementers' Guide in a later version of the ACVC.
7	Test sequence letter (A-Z).
8	Compilation sequence digit (0-9).
9	When there are several compilation units, "M" indicates the main program.

Characters 8 and 9 are only present for tests that consist of several separately compiled units. The eighth character indicates the order in which the units are to be compiled (unit 0 is compiled first). The ninth character is only present for the main program and is always "M". Characters 2-8 uniquely identify a test file within a particular version of the test suite.

The source code of each test file gives its own complete name as the first (comment) line of text. Characters 2-8 suffice to identify a test file uniquely. The maximum length of a test file name is 13 characters (e.g., CA1106A4M.ADA).

Class A tests are expected to compile and execute without error. There are no run-time checks in Class A tests, so theoretically such tests need not be executed; they can only fail execution by crashing. Nonetheless, all Class A tests are executed in a validation attempt.

Class B tests are expected to fail compilation. The lines containing illegal constructs that must be detected at compile time are marked "ERROR:", with a very brief indication of what makes the construct illegal. Class B tests are passed if all illegal constructs can be detected by a compiler, and no legal constructs are rejected. This criteria can be satisfied in several ways. Ideally, all lines marked ERROR will contain an error message identifying the illegal construct, and no other lines will be marked as containing errors. Few compilers can meet this high standard for every test. If there is doubt about whether an illegal construct is being detected, the ultimate criterion is to remove all illegal constructs except the one that is in question. If the compiler rejects a test containing a

single illegal construct, then the compiler is considered to detect the illegality, regardless of the content or placement of its error message(s). Similarly, if a legal construct appears to be rejected for reasons that have nothing to do with other illegal constructs appearing in a test, the only firm evidence of failure to pass the test is to compile a version of the test that contains no illegal constructs.

Class C tests are expected to compile successfully and execute successfully. All Class C tests are self-checking, although some I/O tests require interaction with a terminal. (Such tests have the TYPE name MAN.) C tests use a reporting package that prints out a message indicating what test is being executed, whether it passes or fails the internal check, and if it fails, the nominal reason for the failure. The reporting package is contained in the files REP_SPEC.ADA and REP_BODY.ADA. These files must be compiled before any Class A, C, D, or E test can be successfully compiled or executed.

Class D tests are capacity tests. All Class D tests are executable, and are expected to execute successfully if no errors are reported at compile time. If an error is reported at compile time, it should be an error associated with the capacity being tested. Such "failures" to compile a valid Ada program are not counted during validation, since such D tests are considered "inapplicable" to the implementation. Failure for any other reason, however, may be considered a failure for purposes of validation, and should be reported to the AVO for a ruling.

Class E tests are executable tests that check whether certain implementation-dependent options have been provided or how certain ambiguities in the Standard have been resolved. These tests print out comments giving their results. The pass/fail criteria for Class E tests are test dependent and are indicated in each Class E test.

Class L tests are expected to fail at link time, i.e., an attempt to execute the main program must generate an error message before any declarations in the main program or any units referenced by the main program are elaborated. Such tests need not fail compilation, but may fail to compile for some implementations. In such cases, the source lines containing the illegal construct are marked with an "ERROR:" comment just as for Class B tests.

Tests that have the TYPE name TST are parameterized to accept certain implementation-dependent values. The parameters are given as identifiers whose name begins with a dollar sign ($). The MACRO.DEFS file contains a list of all parameters and a specification of the implementation-dependent value that is to be provided. The appropriate textual value must be substituted for these parameters before the tests are suitable for compilation and/or execution.

All Class A, B, C, and L tests having the TYPE name ADA should be passed by a conforming compiler. Class A, B, C, and L tests having the TYPE name TST should be passed after the appropriate implementation-dependent values have been substituted for parameters in the test. Tests having the TYPE name DEP need only be passed for compilers having certain capabilities, e.g., the compiler purports to support the predefined LONG_FLOAT or LONG_INTEGER data type. Appendix G contains a

list of test applicability criteria that describes the implementation-dependent features of the DEP tests. All DEP tests are executed during a validation attempt except those tests requiring a floating point digits value that exceeds SYSTEM.MAX_DIGITS. DEP tests for a particular construct must all pass or all fail. For example, it would be unacceptable for some LONG_INTEGER DEP tests to pass and some to fail.

APPENDIX A

PREREQUISITES FOR EXECUTING TESTS

In order to run all of the ACVC tests, all of Ada must be implemented, since the tests cover the entire language. However, each test uses only a small portion of Ada®(in addition to the particular Ada feature being tested). In general, we have tried to not use more Ada features than are necessary to write self-checking executable tests that are also not to difficult to read or modify. Thus, we generally stay within a Pascal-like subset of Ada. However, if, for example, we need to use a FOR loop, we use it, rather than use an equivalent WHILE loop or construct a loop with IF and GOTO. In other words, our tests are intended for validating finished implementations rather than for debugging in the earlier stages of implementation.

All of our executable tests use a separately compiled package, REPORT, to automate the reporting of test results and to prevent a compiler from knowing certain values at compile time (and thus optimizing when we don't want it to). The REPORT package uses most of the Ada features that we tend to use throughout the tests, namely

```
type INTEGER, literals, +, -, relations, :=
type CHARACTER, literals, &, relations, :=
type BOOLEAN, literals, AND, OR, NOT, relations, :=
user-defined enumeration type, literals, relations, :=
type STRING, literals, &, relations, :=
object declarations (and initialization) for above types
subprogram declarations, bodies, calls, with formal
   parameters of all modes and of the above types
IF statements
```

Our executable tests generally have the following structure:

```
-- Name.ADA
-- Objective
-- Author
WITH REPORT; USE REPORT;
PROCEDURE Name IS
     ...
BEGIN
     TEST ("Name.ADA", "Description ..." &
                       "...   ...");
```

®Ada is a registered trademark of the U.S. Government Ada Joint Program Office (AJPO).

```
        ...
        IF ... /= CorrectValue THEN
             FAILED ("Reason");
        END IF;
        ...
        RESULT;
    END Name;
```

Hence, at least this much of the REPORT package must be implemented to compile and/or run our executable tests.

Since the REPORT package body is independent of the tests themselves, it can always be recoded to use only those features that are actually working.

We have followed certain coding standards to ensure that the tests are reasonably portable without further modification, for example:

source line length <= 72 characters (except in certain .DEP and .TST tests)
basic 55 character set
small numeric values (12 bit word size is sufficient)
placement of difficult or subsettable features in separate tests
e.g., floating point, fixed point, exceptions, tasking, generics

On the other hand, we have tried to ensure a thorough coverage of Ada and of possible implementations of its features. Thus, for example, if a test objective is concerned with some property of scalar types, then we usually try it for INTEGER, CHARACTER, BOOLEAN, and a user-defined enumeration type (all in one test since we expect all of these types to be implemented); for FLOAT and a user-defined floating point type (in a separate test); and for a fixed point type (in a separate test).

The remaining appendices contain the REPORT package specification and body together with test procedures, CZ1101A.ADA and CZ1102A.ADA, to check that the REPORT package works correctly. If an implementation can successfully compile and execute REPORT, CZ1101A.ADA, and CZ1102A.ADA, then it has a reasonable chance of usefully using the ACVC tests for debugging prior to validation.

Some of the executable I/O tests (tests whose name begins CE...) use a checking program called CHECK_FILE. The source code for this procedure is found in the file CHECK_FILE.ADA. This procedure should be compiled and put in the library that contains the report package. A program to check that CHECK_FILE works is in file CZ1103A.ADA. This program should be executed before executing any I/O tests that use CHECK_FILE. Since CZ1103A checks that errors in text files will be detected, CZ1103A will print out some failure messages when it is run. The presence of these messages does not mean the test has failed.

Some of the executable I/O tests create a file that is to be used by another test. In these instances, the tests must be run in sequence without deleting the file that was created. These are the tests that

create a file, followed by the tests that depend on the file:

CE2108A	must be followed by	CE2108B
CE2108C	must be followed by	CE2108D
CE3112A	must be followed by	CE3112B

APPENDIX B

REPORT SPECIFICATION

```
-- REPORT_SPEC.ADA

-- THIS REPORT PACKAGE PROVIDES THE MECHANISM FOR REPORTING THE
-- PASS/FAIL/NOT-APPLICABLE RESULTS OF EXECUTABLE (CLASSES A, C, D, E,
-- AND L) TESTS.

-- IT ALSO PROVIDES THE MECHANISM FOR GUARANTEEING THAT CERTAIN VALUES
-- BECOME DYNAMIC (NOT KNOWN AT COMPILE-TIME).

-- JRK 12/13/79
-- JRK 6/10/80
-- JRK 8/6/81
-- JRK 10/27/82
-- JRK 6/1/84

PACKAGE REPORT IS

  -- THE REPORT ROUTINES.

     PROCEDURE TEST           -- THIS ROUTINE MUST BE INVOKED AT THE
                              -- START OF A TEST, BEFORE ANY OF THE
                              -- OTHER REPORT ROUTINES ARE INVOKED.
                              -- IT SAVES THE TEST NAME AND OUTPUTS THE
                              -- NAME AND DESCRIPTION.
        ( NAME : STRING;      -- TEST NAME, E.G., "C23001A".
          DESCR : STRING      -- BRIEF DESCRIPTION OF TEST, E.G.,
                              -- "UPPER/LOWER CASE EQUIVALENCE IN " &
                              -- "IDENTIFIERS".
        );

     PROCEDURE FAILED         -- OUTPUT A FAILURE MESSAGE.  SHOULD BE
                              -- INVOKED SEPARATELY TO REPORT THE
                              -- FAILURE OF EACH SUBTEST WITHIN A TEST.
        ( DESCR : STRING      -- BRIEF DESCRIPTION OF WHAT FAILED.
                              -- SHOULD BE PHRASED AS:
                              -- "(FAILED BECAUSE) ...REASON...".
        );

     PROCEDURE NOT_APPLICABLE -- OUTPUT A NOT-APPLICABLE MESSAGE.
```

```
                                -- SHOULD BE INVOKED SEPARATELY TO REPORT
                                -- THE NON-APPLICABILITY OF EACH SUBTEST
                                -- WITHIN A TEST.
        ( DESCR : STRING        -- BRIEF DESCRIPTION OF WHAT IS
                                -- NOT-APPLICABLE. SHOULD BE PHRASED AS:
                                -- "(NOT-APPLICABLE BECAUSE)...REASON...".
        );

    PROCEDURE COMMENT           -- OUTPUT A COMMENT MESSAGE.
        ( DESCR : STRING        -- THE MESSAGE.
        );

    PROCEDURE RESULT;           -- THIS ROUTINE MUST BE INVOKED AT THE
                                -- END OF A TEST.  IT OUTPUTS A MESSAGE
                                -- INDICATING WHETHER THE TEST AS A
                                -- WHOLE HAS PASSED OR FAILED, OR IS
                                -- NOT-APPLICABLE.

-- THE DYNAMIC VALUE ROUTINES.

    -- EVEN WITH STATIC ARGUMENTS, THESE FUNCTIONS WILL HAVE DYNAMIC
    -- RESULTS.

    FUNCTION IDENT_INT          -- AN IDENTITY FUNCTION FOR TYPE INTEGER.
        ( X : INTEGER           -- THE ARGUMENT.
        ) RETURN INTEGER;       -- X.

    FUNCTION IDENT_CHAR         -- AN IDENTITY FUNCTION FOR TYPE
                                -- CHARACTER.
        ( X : CHARACTER         -- THE ARGUMENT.
        ) RETURN CHARACTER;     -- X.

    FUNCTION IDENT_BOOL         -- AN IDENTITY FUNCTION FOR TYPE BOOLEAN.
        ( X : BOOLEAN           -- THE ARGUMENT.
        ) RETURN BOOLEAN;       -- X.

    FUNCTION IDENT_STR          -- AN IDENTITY FUNCTION FOR TYPE STRING.
        ( X : STRING            -- THE ARGUMENT.
        ) RETURN STRING;        -- X.

    FUNCTION EQUAL              -- A RECURSIVE EQUALITY FUNCTION FOR TYPE
                                -- INTEGER.
        ( X, Y : INTEGER        -- THE ARGUMENTS.
        ) RETURN BOOLEAN;       -- X = Y.

END REPORT;
```

APPENDIX C

REPORT BODY

```
-- REPORT_BODY.ADA

-- DCB 04/27/80
-- JRK 6/10/80
-- JRK 11/12/80
-- JRK 8/6/81
-- JRK 10/27/82
-- JRK 6/1/84
-- JRK 11/18/85  ADDED PRAGMA ELABORATE.

WITH TEXT_IO;
USE TEXT_IO;
PRAGMA ELABORATE (TEXT_IO);

PACKAGE BODY REPORT IS

     TYPE STATUS IS (PASS, FAIL, DOES_NOT_APPLY);
     TEST_STATUS : STATUS := FAIL;

     NO_NAME : CONSTANT STRING (1..7) := "NO_NAME";
     MAX_NAME_LEN : CONSTANT := 15;      -- MAXIMUM TEST NAME LENGTH.
     TEST_NAME_LEN : INTEGER RANGE 0..MAX_NAME_LEN := 0;
     TEST_NAME : STRING (1..MAX_NAME_LEN);

     PROCEDURE PUT_MSG (MSG : STRING) IS
          -- WRITE MESSAGE.  LONG MESSAGES ARE FOLDED (AND INDENTED).
          MAX_LEN : CONSTANT INTEGER RANGE 50..150 := 72;  -- MAXIMUM
                                        -- OUTPUT LINE LENGTH.
          INDENT : CONSTANT INTEGER := TEST_NAME_LEN + 9;  -- AMOUNT TO
                                        -- INDENT CONTINUATION LINES.
          I : INTEGER := 0;             -- CURRENT INDENTATION.
          M : INTEGER := MSG'FIRST;     -- START OF MESSAGE SLICE.
          N : INTEGER;                  -- END OF MESSAGE SLICE.
     BEGIN
          LOOP
               IF I + (MSG'LAST-M+1) > MAX_LEN THEN
                    N := M + (MAX_LEN-I) - 1;
                    IF MSG (N) /= ' ' THEN
                         WHILE N >= M AND THEN MSG (N+1) /= ' ' LOOP
                              N := N - 1;
```

```
                    END LOOP;
                    IF N < M THEN
                         N := M + (MAX_LEN-I) - 1;
                    END IF;
               END IF;
          ELSE N := MSG'LAST;
          END IF;
          SET_COL (STANDARD_OUTPUT, COUNT (I+1));
          PUT_LINE (STANDARD_OUTPUT, MSG (M..N));
          I := INDENT;
          M := N + 1;
          WHILE M <= MSG'LAST AND THEN MSG (M) = ' ' LOOP
               M := M + 1;
          END LOOP;
          EXIT WHEN M > MSG'LAST;
     END LOOP;
END PUT_MSG;

PROCEDURE TEST (NAME : STRING; DESCR : STRING) IS
BEGIN
     TEST_STATUS := PASS;
     IF NAME'LENGTH <= MAX_NAME_LEN THEN
          TEST_NAME_LEN := NAME'LENGTH;
     ELSE TEST_NAME_LEN := MAX_NAME_LEN;
     END IF;
     TEST_NAME (1..TEST_NAME_LEN) :=
               NAME (NAME'FIRST .. NAME'FIRST+TEST_NAME_LEN-1);
     PUT_MSG ("");
     PUT_MSG ("---- " & TEST_NAME (1..TEST_NAME_LEN) & " " &
               DESCR & ".");
END TEST;

PROCEDURE COMMENT (DESCR : STRING) IS
BEGIN
     PUT_MSG ("   - " & TEST_NAME (1..TEST_NAME_LEN) & " " &
               DESCR & ".");
END COMMENT;

PROCEDURE FAILED (DESCR : STRING) IS
BEGIN
     TEST_STATUS := FAIL;
     PUT_MSG ("   * " & TEST_NAME (1..TEST_NAME_LEN) & " " &
               DESCR & ".");
END FAILED;

PROCEDURE NOT_APPLICABLE (DESCR : STRING) IS
BEGIN
     IF TEST_STATUS = PASS THEN
          TEST_STATUS := DOES_NOT_APPLY;
     END IF;
     PUT_MSG ("   + " & TEST_NAME (1..TEST_NAME_LEN) & " " &
               DESCR & ".");
END NOT_APPLICABLE;

PROCEDURE RESULT IS
```

```
BEGIN
     IF TEST_STATUS = PASS THEN
          PUT_MSG ("==== " & TEST_NAME (1..TEST_NAME_LEN) &
                   " PASSED ====================.");
     ELSIF TEST_STATUS = DOES_NOT_APPLY THEN
          PUT_MSG ("++++ " & TEST_NAME (1..TEST_NAME_LEN) &
                   " NOT-APPLICABLE ++++++++++++++++++++.");
     ELSE PUT_MSG ("**** " & TEST_NAME (1..TEST_NAME_LEN) &
                   " FAILED ********************.");
     END IF;
     TEST_STATUS := FAIL;
     TEST_NAME_LEN := NO_NAME'LENGTH;
     TEST_NAME (1..TEST_NAME_LEN) := NO_NAME;
END RESULT;

FUNCTION IDENT_INT (X : INTEGER) RETURN INTEGER IS
BEGIN
     IF EQUAL (X, X) THEN          -- ALWAYS EQUAL.
          RETURN X;                -- ALWAYS EXECUTED.
     END IF;
     RETURN 0;                     -- NEVER EXECUTED.
END IDENT_INT;

FUNCTION IDENT_CHAR (X : CHARACTER) RETURN CHARACTER IS
BEGIN
     IF EQUAL (CHARACTER'POS(X), CHARACTER'POS(X)) THEN  -- ALWAYS
                                   -- EQUAL.
          RETURN X;                -- ALWAYS EXECUTED.
     END IF;
     RETURN '0';                   -- NEVER EXECUTED.
END IDENT_CHAR;

FUNCTION IDENT_BOOL (X : BOOLEAN) RETURN BOOLEAN IS
BEGIN
     IF EQUAL (BOOLEAN'POS(X), BOOLEAN'POS(X)) THEN  -- ALWAYS
                                   -- EQUAL.
          RETURN X;                -- ALWAYS EXECUTED.
     END IF;
     RETURN FALSE;                 -- NEVER EXECUTED.
END IDENT_BOOL;

FUNCTION IDENT_STR (X : STRING) RETURN STRING IS
BEGIN
     IF EQUAL (X'LENGTH, X'LENGTH) THEN  -- ALWAYS EQUAL.
          RETURN X;                -- ALWAYS EXECUTED.
     END IF;
     RETURN "";                    -- NEVER EXECUTED.
END IDENT_STR;

FUNCTION EQUAL (X, Y : INTEGER) RETURN BOOLEAN IS
     REC_LIMIT : CONSTANT INTEGER RANGE 1..100 := 3;  -- RECURSION
                                   -- LIMIT.
     Z : BOOLEAN;                  -- RESULT.
BEGIN
     IF X < 0 THEN
```

```
                IF Y < 0 THEN
                     Z := EQUAL (-X, -Y);
                ELSE Z := FALSE;
                END IF;
           ELSIF X > REC_LIMIT THEN
                Z := EQUAL (REC_LIMIT, Y-X+REC_LIMIT);
           ELSIF X > 0 THEN
                Z := EQUAL (X-1, Y-1);
           ELSE Z := Y = 0;
           END IF;
           RETURN Z;
      EXCEPTION
           WHEN OTHERS =>
                RETURN X = Y;
      END EQUAL;

BEGIN

      TEST_NAME_LEN := NO_NAME'LENGTH;
      TEST_NAME (1..TEST_NAME_LEN) := NO_NAME;

END REPORT;
```

APPENDIX D

REPORT ACCEPTANCE TEST

```
-- CZ1101A.ADA

-- CHECK THAT THE REPORT ROUTINES OF THE REPORT PACKAGE WORK CORRECTLY.

-- JRK 8/7/81
-- JRK 10/27/82
-- JRK 6/1/84

WITH REPORT;
USE REPORT;

PROCEDURE CZ1101A IS

BEGIN

     COMMENT ("CZ1101A: CHECK REPORT ROUTINES. CHECKING 'COMMENT'");

     COMMENT ("CHECKING 'PUT_MSG' FOR PROPER OUTPUT");
     COMMENT ("THIS LINE IS EXACTLY 'MAX_LEN' LONG. " &
              "....5...60....5...70.");
     COMMENT ("THIS COMMENT HAS A WORD THAT SPANS THE FOLD " &
              "POINT. THIS COMMENT FITS EXACTLY ON TWO LINES. " &
              "...5...60....5...70.");
     COMMENT ("THIS COMMENT HAS A WORD THAT ENDS AT THE " &
              "FOLD POINT ...70..     THE SPACES JUST AFTER " &
              "THE FOLD POINT SHOULD BE GONE");
     COMMENT ("THIS COMMENT HAS A WORD THAT BEGINS AT THE " &
              "FOLD POINT.     THIS SHOULD BE ON THE SECOND LINE");
     COMMENT ("THIS COMMENT HAS SPACES SPANNING THE FOLD " &
              "POINT. ..5...70     THIS SHOULD BE ON THE " &
              "SECOND LINE");
     COMMENT ("THIS_COMMENT_IS_ONE_VERY_LONG_WORD_AND_SO_" &
              "IT_SHOULD_BE_SPLIT_AT_THE_FOLD_POINT");

     COMMENT ("CHECKING THAT REPORT BODY INITIALIZES TO " &
              "'NO_NAME' AND 'FAILED'. CHECKING 'RESULT'");
     RESULT;

     TEST ("PASS_TEST", "CHECKING 'TEST' AND 'RESULT' FOR 'PASSED'");
```

```
        COMMENT ("CHECKING THAT INDENTATION OF CONTINUATION LINES IS " &
                 "ADJUSTED ACCORDING TO TEST NAME LENGTH");
        RESULT;

        COMMENT ("CHECKING 'RESULT' FOR RESETTING TO 'NO_NAME' " &
                 "AND 'FAILED'");
        RESULT;
        COMMENT ("CHECKING THAT INDENTATION OF CONTINUATION LINES IS " &
                 "ADJUSTED ACCORDING TO TEST NAME LENGTH");

        TEST ("FAIL_TEST", "CHECKING 'FAILED' AND 'RESULT' FOR 'FAILED'");
        FAILED ("CHECKING 'FAILED'. NOW 'RESULT' SHOULD BE 'FAILED'");
        RESULT;

        TEST ("NA_TEST", "CHECKING 'NOT_APPLICABLE' AND 'RESULT' FOR " &
                         "'NOT-APPLICABLE'");
        NOT_APPLICABLE ("CHECKING 'NOT_APPLICABLE'. NOW 'RESULT' SHOULD " &
                        "BE 'NOT-APPLICABLE'");
        RESULT;

        TEST ("FAIL_NA_TEST", "CHECKING 'FAILED', 'NOT_APPLICABLE', AND " &
                              "'RESULT' FOR 'FAILED'");
        NOT_APPLICABLE ("CHECKING 'NOT_APPLICABLE'. NOW 'RESULT' SHOULD " &
                        "BE 'NOT-APPLICABLE'");
        FAILED ("CHECKING 'FAILED'. NOW 'RESULT' SHOULD BE 'FAILED'");
        NOT_APPLICABLE ("CHECKING 'NOT_APPLICABLE'. NOW 'RESULT' SHOULD " &
                        "STAY 'FAILED'");
        RESULT;

        COMMENT ("END CZ1101A");

    END CZ1101A;


    -- CZ1102A.ADA

    -- CHECK THAT THE DYNAMIC VALUE ROUTINES OF THE REPORT PACKAGE WORK
    --   CORRECTLY.

    -- JRK 8/7/81
    -- JRK 10/27/82

    WITH REPORT;
    USE REPORT;

    PROCEDURE CZ1102A IS

    BEGIN

        TEST ("CZ1102A", "CHECK THAT THE DYNAMIC VALUE ROUTINES OF " &
                         "THE REPORT PACKAGE WORK CORRECTLY");

        IF NOT EQUAL (0, 0) OR
           EQUAL (0, 1) OR
           NOT EQUAL (1, 1) OR
```

```
           NOT EQUAL (3, 3) OR
           NOT EQUAL (4, 4) OR
           NOT EQUAL (-1, -1) OR
           EQUAL (-1, 0) THEN
             FAILED ("'EQUAL' NOT WORKING");
        END IF;

        IF IDENT_INT (5) /= 5 THEN
             FAILED ("'IDENT_INT' NOT WORKING");
        END IF;

        IF IDENT_CHAR ('E') /= 'E' THEN
             FAILED ("'IDENT_CHAR' NOT WORKING");
        END IF;

        IF IDENT_BOOL (TRUE) /= TRUE THEN
             FAILED ("'IDENT_BOOL' NOT WORKING");
        END IF;

        IF IDENT_STR ("") /= "" OR
           IDENT_STR ("K") /= "K" OR
           IDENT_STR ("PQRS") /= "PQRS" THEN
             FAILED ("'IDENT_STR' NOT WORKING");
        END IF;

        RESULT;

    END CZ1102A;
```

APPENDIX E

MACRO DEFINITIONS

```
-- MACRO.DEFS

-- THIS FILE CONTAINS THE MACRO DEFINITIONS USED IN THE ACVC TESTS.
-- THESE DEFINITIONS ARE FOR THE PRELIMINARY ADA TEST TRANSLATOR.


-- THE MAXIMUM INPUT LINE LENGTH PERMITTED BY THE COMPILER FOR ADA
-- SOURCE CODE.
-- USED IN: COMMENTS IN THIS FILE, A26007A
MAX_IN_LEN      120

-- TWO IDENTIFIERS THAT ARE MAX_IN_LEN LONG AND DIFFER ONLY IN THEIR
-- LAST CHARACTER.
-- USED IN: C23003A, C23003B, C23003C, B23003D, B23003E, B23003F,
--          C23003G, C23003H, C23003I, C23003J, C35502D, C35502F
BIG_ID1 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA1
BIG_ID2 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA2

-- TWO IDENTIFIERS THAT ARE MAX_IN_LEN LONG AND DIFFER ONLY IN THEIR
-- MIDDLE CHARACTER.
-- USED IN: C23003A, C23003B, C23003C, C23003G, C23003H, C23003I,
--          C23003J
BIG_ID3 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA3AAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BIG_ID4 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA4AAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

-- TWO STRING LITERALS WHOSE CONCATENATION IS THE IMAGE OF BIG_ID1.
-- USED IN: C35502D, C35502F
BIG_STRING1     "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"
BIG_STRING2     "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA1"

-- A STRING LITERAL THAT IS MAX_IN_LEN CHARACTERS LONG (INCLUDING THE
-- QUOTE CHARACTERS).
-- USED IN: A26007A
MAX_STRING_LITERAL      "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"
```

```
-- A BASED_INTEGER LITERAL (PREFERABLY BASE 8 OR 16) WHOSE HIGHEST ORDER
-- NON-ZERO BIT WOULD FALL IN THE SIGN BIT POSITION OF THE
-- REPRESENTATION FOR SYSTEM.MAX_INT, I.E., AN ATTEMPT TO WRITE A
-- NEGATIVE VALUED LITERAL SUCH AS -2.
-- USED IN: E24101A
NEG_BASED_INT   16#FFFFFFFE#

-- A UNIVERSAL_INTEGER LITERAL WHOSE VALUE IS 298, BUT WITH ENOUGH
-- LEADING ZEROES SO THAT IT IS MAX_IN_LEN CHARACTERS LONG.
-- USED IN: C24003A
BIG_INT_LIT     0000000000000000000000000000000000000000000000000000000000000000
                0000000000000000000000000000000000000000000000000000000000000298

-- A UNIVERSAL_REAL LITERAL WHOSE VALUE IS 690.0, BUT WITH ENOUGH
-- LEADING ZEROES SO THAT IT IS MAX_IN_LEN CHARACTERS LONG.
-- USED IN: C24003B, C24003C
BIG_REAL_LIT    0000000000000000000000000000000000000000000000000000000000000000
                0000000000000000000000000000000000000000000000000000000000690.0

-- A UNIVERSAL_INTEGER BASED LITERAL (USING COLONS) WHOSE VALUE IS
-- 2:11:, BUT WITH ENOUGH LEADING ZEROES IN THE MANTISSA SO THAT THE
-- LITERAL IS MAX_IN_LEN CHARACTERS LONG.
-- USED IN: C2A009A
MAX_LEN_INT_BASED_LITERAL       2:0000000000000000000000000000000000000000000
                                  0000000000000000000000000000000000000000000
                                  000000000000000000000000000000011:
-- A UNIVERSAL_REAL BASED LITERAL (USING COLONS) WHOSE VALUE IS
-- 16:F.E:, BUT WITH ENOUGH LEADING ZEROES IN THE MANTISSA SO THAT THE
-- LITERAL IS MAX_IN_LEN CHARACTERS LONG.
-- USED IN: C2A009A
MAX_LEN_REAL_BASED_LITERAL      16:0000000000000000000000000000000000000000000
                                  0000000000000000000000000000000000000000000
                                  000000000000000000000000000000F.E:

-- A SEQUENCE OF BLANKS THAT IS MAX_IN_LEN - 20 LONG.
-- USED IN: B22001A, B22001B, B22001C, B22001D, B22001E, B22001F,
--          B22001G, B22001I, B22001J, B22001K, B22001L, B22001M,
--          B22001N
BLANKS

-- THE UNIVERSAL_INTEGER LITERAL WHOSE VALUE IS SYSTEM.MAX_DIGITS.
-- USED IN: B35701A
MAX_DIGITS      33

-- A NAME OF A PREDEFINED NUMERIC TYPE OTHER THAN FLOAT, INTEGER,
-- SHORT_FLOAT, SHORT_INTEGER, LONG_FLOAT, OR LONG_INTEGER.
-- (MOST IMPLEMENTATIONS WILL NOT HAVE ANY SUCH TYPES.)
-- USED IN: C45231D, B86001DT
NAME    SHORT_SHORT_INTEGER

-- THE (SIGNED) UNIVERSAL_INTEGER LITERALS WHOSE VALUES ARE
-- INTEGER'FIRST AND INTEGER'LAST.
-- USED IN: C35503F, C45232A, B54B01B
INTEGER_FIRST   -2_147_483_648
INTEGER_LAST    2_147_483_647
```

```
-- THE UNIVERSAL_INTEGER LITERAL WHOSE VALUE IS INTEGER'LAST + 1.
-- USED IN: C45232A
INTEGER_LAST_PLUS_1     2_147_483_648

-- THE (SIGNED) UNIVERSAL_INTEGER LITERAL WHOSE VALUE IS SYSTEM.MIN_INT.
-- USED IN: C35503D, C35503F
MIN_INT -2147483648

-- THE UNIVERSAL_INTEGER LITERAL WHOSE VALUE IS SYSTEM.MAX_INT.
-- USED IN: C35503D, C35503F, C4A007A
MAX_INT 2147483647

-- THE UNIVERSAL_INTEGER LITERAL WHOSE VALUE IS SYSTEM.MAX_INT + 1.
-- USED IN: C45232A
MAX_INT_PLUS_1  2_147_483_648

-- A (SIGNED) UNIVERSAL_REAL LITERAL WHOSE VALUE (NOT SUBJECT TO
-- ROUND-OFF ERROR IF POSSIBLE) LIES BETWEEN DURATION'BASE'FIRST AND
-- DURATION'FIRST.  IF NO SUCH VALUES EXIST, ANY VALUE IN THE RANGE OF
-- DURATION WILL DO.
-- USED IN: C96005B
LESS_THAN_DURATION      -75_000.0

-- A UNIVERSAL_REAL LITERAL WHOSE VALUE (NOT SUBJECT TO ROUND-OFF ERROR
-- IF POSSIBLE) LIES BETWEEN DURATION'BASE'LAST AND DURATION'LAST.  IF
-- NO SUCH VALUES EXIST, ANY VALUE IN THE RANGE OF DURATION WILL DO.
-- USED IN: C96005B
GREATER_THAN_DURATION   75_000.0

-- (SIGNED) UNIVERSAL_REAL LITERALS WHOSE VALUES ARE LESS THAN
-- DURATION'BASE'FIRST AND GREATER THAN DURATION'BASE'LAST.
-- USED IN: C96005C
LESS_THAN_DURATION_BASE_FIRST   -131_073.0
GREATER_THAN_DURATION_BASE_LAST 131_073.0

-- THE UNIVERSAL_INTEGER LITERAL WHOSE VALUE IS TEXT_IO.COUNT'LAST.
-- USED IN: CE3002B
COUNT_LAST      2_147_483_647

-- THE UNIVERSAL_INTEGER LITERAL WHOSE VALUE IS TEXT_IO.FIELD'LAST.
-- USED IN: CE3002C
FIELD_LAST      2_147_483_647

-- AN ILLEGAL EXTERNAL FILE NAME THAT EITHER (PREFERABLY) CONTAINS
-- INVALID CHARACTERS OR IS TOO LONG.
-- USED IN: CE2102C, CE2102H
FILE_NAME_WITH_BAD_CHARS        BAD-CHARS^#.%!X

-- AN EXTERNAL FILE NAME THAT EITHER (PREFERABLY) CONTAINS A WILD CARD
-- CHARACTER OR IS TOO LONG.
-- USED IN: CE2102C, CE2102H
FILE_NAME_WITH_WILD_CARD_CHAR   WILD-CHAR*.NAM

-- AN ILLEGAL EXTERNAL FILE NAME (E.G., TOO LONG, OR CONTAINING INVALID
```

```
-- CHARACTERS.
-- USED IN: CE2103A, CE2103B, CE3102B, CE3107A
ILLEGAL_EXTERNAL_FILE_NAME1     BADCHAR^@.~!
ILLEGAL_EXTERNAL_FILE_NAME2     THIS-FILE-NAME-WOULD-BE-PERFECTLY-LEGAL-IF-
                                IT-WERE-NOT-SO-LONG--IT-HAS-NEARLY-ONE-HUNDRED-
                                SIXTY-CHARACTERS-WHICH-MAKES-IT-MUCH-TOO-LONG-
                                FOR-VMS.SO-THERE
```

APPENDIX F

NON-STANDARD TEST NAMES

C900ACA.ADA C910BDA.ADA C910BDB.ADA C950CHC.ADA

APPENDIX G

TEST APPLICABILITY CRITERIA

Certain tests may be inapplicable (N/A) according to the way an implementation treats certain language features. These implementation choices and the tests affected are listed in this section. Note that N/A tests are still to be compiled (and executed if not of Class B) to verify the implementation's treatment of these features.

- Is SHORT_INTEGER supported? (Y/N) _____

  If No, the following tests are inapplicable:

  B52004E.DEP
  B55B09D.DEP
  C45231B.DEP
  C45304B.DEP
  C45502B.DEP
  C45503B.DEP
  C45504B.DEP
  C45504E.DEP
  C45611B.DEP
  C45613B.DEP
  C45614B.DEP
  C45631B.DEP
  C45632B.DEP
  C55B07B.DEP
  B86001CR.DEP

- Is LONG_INTEGER supported? (Y/N) _____

  If No, the following tests are inapplicable:

  B52004D.DEP
  B55B09C.DEP
  C45231C.DEP
  C45304C.DEP
  C45502C.DEP
  C45503C.DEP
  C45504C.DEP
  C45504F.DEP
  C45611C.DEP
  C45613C.DEP
  C45614C.DEP
  C45631C.DEP
  C45632C.DEP
  C55B07A.DEP
  B86001CS.DEP

- Is SHORT_FLOAT supported? (Y/N) _____

  If No, the following tests are inapplicable:

  C35702A.DEP
  B86001CP.DEP

- Is LONG_FLOAT supported? (Y/N) _____

  If No, the following tests are inapplicable:

  C35702B.DEP
  B86001CQ.DEP

- Are representation specifications for non-contiguous values allowed? (Y/N) _____

  If No, the following tests are inapplicable:

  A39005F.DEP
  C35502I.DEP
  C35502J.DEP
  C35502M.DEP
  C35502N.DEP
  C35507J.DEP
  C35507M.DEP
  C35507N.DEP
  C35508I.DEP
  C35508J.DEP
  C35508M.DEP
  C35508N.DEP
  C55B16A.DEP

- Is some numeric type other than INTEGER, SHORT_INTEGER, LONG_INTEGER, FLOAT, SHORT_FLOAT, and LONG_FLOAT supported? (Y/N) _____

  If No, the following tests are inapplicable:

  B86001DT.TST
  C45231D.TST

- Is the package SYSTEM used by package TEXT_IO? (Y/N) _____

  If Yes, the following tests are inapplicable:

  C86001F.DEP

- Are 'SIZE representation clauses for enumeration types supported (Y/N)? _____

  If No, the following test is inapplicable:

  A39005B.DEP

- Are 'SIZE representation clauses for derived integer types supported (Y/N)? _____

  If No, the following test is inapplicable:

  C87B62A.DEP

- Are 'STORAGE_SIZE representation clauses for access types supported (Y/N)? _____

  If No, the following tests are inapplicable:

  A39005C.DEP
  C87B62B.DEP

- Are 'STORAGE_SIZE representation clauses for task types supported (Y/N)? _____

  If No, the following tests are inapplicable:

  A39005D.DEP
  C87B62D.DEP

- Is the 'SMALL clause supported (Y/N)? _____

  If No, the following tests are inapplicable:

  A39005E.DEP
  C87B62C.DEP

- Are enumeration representation clauses supported for enumeration types other than character and boolean types? (Y/N) _____

  If No, the following tests are inapplicable:

  A39005F.DEP C35502I.DEP
  C35502J.DEP C35502M.DEP
  C35502N.DEP

- Are enumeration representation clauses supported for character types? (Y/N) _____

  If No, the following tests are inapplicable:

  C35507I.DEP C35507J.DEP
  C35507M.DEP C35507N.DEP
  C55B16A.DEP

- Are enumeration representation clauses (with representation values other than (FALSE => 0, TRUE =>1)) supported for boolean types? (Y/N) _____

  If No, the following tests are inapplicable:

  C35508I.DEP C35508J.DEP
  C35508M.DEP C35508N.DEP

- Are record representation clauses supported? (Y/N) _____

  If No, the following test is inapplicable:

  A39005G.DEP

- Can generic library subprogram bodies be compiled in separate files from the specifications? (Y/N) _____

  If No, the following test is inapplicable:

  CA1012A*.DEP

- Can a generic library package body be compiled in a different file from its specification? (Y/N) _____

  If No, then the following tests are inapplicable:

  BC3204C*.DEP
  BC3204D*.DEP

- Can a generic non-library package body be compiled as a subunit in a separate file from its specification? (Y/N) _____

  If No, the following test is inapplicable:

  CA2009C*.DEP

- Can a generic non-library subprogram body be compiled as a subunit in a separate file from its specification? (Y/N) _____

  If No, then the following test is inapplicable:

  CA2009F*.DEP

- Can a generic procedure have subunits which are compiled in separate files from the generic unit? (Y/N) _____

  If No, then the following test is inapplicable:

  CA3011A*.DEP

- Can SEQUENTIAL_IO be instantiated with unconstrained array types? (Y/N) _____

  If No, then the following tests are inapplicable:

  AE2101C.DEP
  EE2201D.ADA

- Can SEQUENTIAL_IO be instantiated with record types with discriminants with no default values? (Y/N) _____

  If No, then the following tests are inapplicable:

  AE2101C.DEP
  EE2201E.ADA

- Can DIRECT_IO be instantiated with unconstrained array types? (Y/N) _____

  If No, then the following tests are inapplicable:

  AE2101H.DEP
  EE2401D.ADA

- Can DIRECT_IO be instantiated with record types with discriminants with no default values? (Y/N) _____

  If No, then the following tests are inapplicable:

  AE2101H.DEP
  EE2401G.ADA

- Are there any strings which are illegal as external names of files? (Y/N) _____

  If No, then the following tests are inapplicable:

  CE2102C.TST
  CE2102H.TST

- Consider the following declarations:

```
TYPE F IS DIGITS SYSTEM.MAX_DIGITS;
N : CONSTANT := 2.0 * F'MACHINE_RADIX ** F'MACHINE_EMAX;
```

  The following test is inapplicable if the compiler rejects the declaration of N on the grounds that evaluation of the expression would raise an exception:

  C4A013B.ADA

- Does the implementation support the following three fixed point type definitions? (Y/N) _____

```
DELTA 0.5 RANGE -2.0**10 .. 2.0**10 - 0.5
DELTA 1.0 RANGE -2.0**11 .. 2.0**11 - 1.0
DELTA 2.0**10 RANGE -2.0**21 .. 2.0**21 - 2.0**10
```

  If No, then the following tests are inapplicable:

  C45531C.DEP
  C45531D.DEP
  C45532C.DEP
  C45532D.DEP

- Does the implementation support the following three fixed point type definitions? (Y/N) _____

```
DELTA 0.5 RANGE -2.0**14 .. 2.0**14 - 0.5
DELTA 1.0 RANGE -2.0**15 .. 2.0**15 - 1.0
DELTA 2.0**14 RANGE -2.0**29 .. 2.0**29 - 2.0**14
```

If No, then the following tests are inapplicable:

C45531G.DEP
C45531H.DEP
C45532G.DEP
C45532H.DEP

- Does the implementation support the following three fixed point type definitions? (Y/N) _____

```
DELTA 0.5 RANGE -2.0**30 .. 2.0**30 - 0.5
DELTA 1.0 RANGE -2.0**31 .. 2.0**31 - 1.0
DELTA 2.0**30 RANGE -2.0**61 .. 2.0**61 - 2.0**30
```

If No, then the following tests are inapplicable:

C45531K.DEP
C45531L.DEP
C45532K.DEP
C45532L.DEP

- Does the implementation support the following three fixed point type definitions? (Y/N) _____

```
DELTA 0.5 RANGE -2.0**46 .. 2.0**46 - 0.5
DELTA 1.0 RANGE -2.0**47 .. 2.0**47 - 1.0
DELTA 2.0**46 RANGE -2.0**93 .. 2.0**93 - 2.0**46
```

If No, then the following tests are inapplicable:

C455310.DEP
C45531P.DEP
C455320.DEP
C45532P.DEP

- Does the implementation support the following three fixed point type definitions? (Y/N) _____

```
DELTA 2.0**-12 RANGE -0.5 .. 0.5 - 2.0**-12
DELTA 2.0**-11 RANGE -1.0 .. 1.0 - 2.0**-11
DELTA 2.0**-10 RANGE -2.0 .. 2.0 - 2.0**-10
```

If No, then the following tests are inapplicable:

```
C45531A.DEP
C45531B.DEP
C45532A.DEP
C45532B.DEP
```

- Does the implementation support the following three fixed point type definitions? (Y/N) _____

```
DELTA 2.0**-16 RANGE -0.5 .. 0.5 - 2.0**-16
DELTA 2.0**-15 RANGE -1.0 .. 1.0 - 2.0**-15
DELTA 2.0**-14 RANGE -2.0 .. 2.0 - 2.0**-14
```

If No, then the following tests are inapplicable:

```
C45531E.DEP
C45531F.DEP
C45532E.DEP
C45532F.DEP
```

- Does the implementation support the following three fixed point type definitions? (Y/N) _____

```
DELTA 2.0**-32 RANGE -0.5 .. 0.5 - 2.0**-32
DELTA 2.0**-31 RANGE -1.0 .. 1.0 - 2.0**-31
DELTA 2.0**-30 RANGE -2.0 .. 2.0 - 2.0**-30
```

If No, then the following tests are inapplicable:

```
C45531I.DEP
C45531J.DEP
C45532I.DEP
C45532J.DEP
```

- Does the implementation support the following three fixed point type definitions? (Y/N) _____

```
DELTA 2.0**-48 RANGE -0.5 .. 0.5 - 2.0**-48
DELTA 2.0**-47 RANGE -1.0 .. 1.0 - 2.0**-47
DELTA 2.0**-46 RANGE -2.0 .. 2.0 - 2.0**-46
```

If No, then the following tests are inapplicable:

C45531M.DEP
C45531N.DEP
C45532M.DEP
C45532N.DEP

o What is the value of SYSTEM.MAX_DIGITS? _____

The following tests depend on the value of SYSTEM.MAX_DIGITS:

C24113A,B,...Y
C35705A,B,...Y
C35706A,B,...Y
C35707A,B,...Y
C35708A,B,...Y
C35802A,B,...Y
C45241A,B,...Y
C45321A,B,...Y
C45421A,B,...Y
C45521A,B,...Z
C45524A,B,...Z
C45621A,B,...Z
C45641A,B,...Y
C46012A,B,...Z

Use the table below to determine which of these tests are applicable based on the value of SYSTEM.MAX_DIGITS.

MAX_DIGITS	Last Letter of Test Name
5	A
6	A,B
7	A..C
8	A..D
9	A..E
10	A..F
11	A..G
12	A..H
13	A..I
14	A..J
15	A..K
16	A..L
17	A..M
18	A..N
19	A..O
20	A..P
21	A..Q
22	A..R
23	A..S
24	A..T
25	A..U
26	A..V
27	A..W
28	A..X
29	A..Y
30	A..Z

APPENDIX H

EXECUTABLE TESTS WHICH MAY BE INAPPLICABLE

Some Class C and Class E tests can detect,at run-time, certain implementation characteristics which make the test inapplicable. The result reported for these tests is NOT-APPLICABLE. These tests must be compiled and executed. They are listed here, grouped according to the implementation characteristics which make them inapplicable.

There are also certain implementation characteristics which the tests do not detect at compile/link-time, but which may not report NOT-APPLICABLE, even though they are ruled inapplicable to the implementation. These tests are given at the end of this Appendix.

TESTS WHICH SHOULD REPORT "INAPPLICABLE"

- Is USE_ERROR or NAME_ERROR raised by every attempt to create or open a text file? (Y/N) _____ (This is the appropriate behavior for an implementation which does not support text files other than the standard input and output. See AI-00332.)

  If Yes, then the following tests should report NOT-APPLICABLE:

CE2109C.ADA	CE3102B.ADA	EE3102C.ADA
CE3103A.ADA	CE3104A.ADA	CE3107A.ADA
CE3108A.ADA	CE3108B.ADA	CE3109A.ADA
CE3110A.ADA	CE3111A.ADA	CE3111B.ADA
CE3111C.ADA	CE3111D.ADA	CE3111E.ADA
CE3112A.ADA	CE3112B.ADA	CE3114A.ADA
CE3114B.ADA	CE3115A.ADA	CE3203A.ADA
CE3208A.ADA	CE3301A.ADA	CE3301B.ADA
CE3301C.ADA	CE3302A.ADA	CE3305A.ADA
CE3402A.ADA	CE3402B.ADA	CE3402C.ADA
CE3402D.ADA	CE3403A.ADA	CE3403B.ADA
CE3403C.ADA	CE3403E.ADA	CE3403F.ADA
CE3404A.ADA	CE3404B.ADA	CE3404C.ADA
CE3405A.ADA	CE3405B.ADA	CE3405C.ADA
CE3405D.ADA	CE3406A.ADA	CE3406B.ADA
CE3406C.ADA	CE3406D.ADA	CE3407A.ADA
CE3407B.ADA	CE3407C.ADA	CE3408A.ADA
CE3408B.ADA	CE3408C.ADA	CE3409A.ADA
CE3409C.ADA	CE3409D.ADA	CE3409E.ADA
CE3409F.ADA	CE3410A.ADA	CE3410C.ADA
CE3410D.ADA	CE3410E.ADA	CE3410F.ADA
CE3411A.ADA	CE3412A.ADA	CE3413A.ADA
CE3413C.ADA	CE3419A.ADA	CE3602A.ADA
CE3602B.ADA	CE3602C.ADA	CE3602D.ADA
CE3603A.ADA	CE3604A.ADA	CE3605A.ADA
CE3605B.ADA	CE3605C.ADA	CE3605D.ADA
CE3605E.ADA	CE3606A.ADA	CE3606B.ADA
CE3704A.ADA	CE3704B.ADA	CE3704D.ADA
CE3704E.ADA	CE3704F.ADA	CE3704M.ADA
CE3704N.ADA	CE3704O.ADA	CE3706D.ADA
CE3706F.ADA	CE3804A.ADA	CE3804B.ADA
CE3804C.ADA	CE3804D.ADA	CE3804E.ADA
CE3804G.ADA	CE3804I.ADA	CE3804K.ADA
CE3804M.ADA	CE3805A.ADA	CE3805B.ADA
CE3806A.ADA	CE3806D.ADA	CE3806E.ADA
CE3905A.ADA	CE3905B.ADA	CE3905C.ADA
CE3905L.ADA	CE3906A.ADA	CE3906B.ADA
CE3906C.ADA	CE3906E.ADA	CE3906F.ADA

- Is USE_ERROR or NAME_ERROR raised by every attempt to create or open a sequential file? (Y/N) _____ (This is the appropriate behavior for an implementation which does not support sequential files. See AI-00332.)

  If Yes, then the following tests should report NOT-APPLICABLE:

CE2102C.TST	CE2104A.ADA	CE2104B.ADA
CE2105A.ADA	CE2106A.ADA	CE2107A.ADA
CE2107B.ADA	CE2107C.ADA	CE2107D.ADA
CE2108A.ADA	CE2108B.ADA	CE2109A.ADA
CE2110A.ADA	CE2110C.ADA	CE2111A.ADA
CE2111C.ADA	CE2111D.ADA	CE2115B.ADA
CE2201A.ADA	CE2201B.ADA	CE2201C.ADA
EE2201D.ADA	EE2201E.ADA	CE2201F.ADA
CE2201G.ADA	CE2204A.ADA	CE2204B.ADA
CE2208B.ADA	CE2210A.ADA	

- Is USE_ERROR or NAME_ERROR raised by every attempt to create or open a direct access file? (Y/N) _____ (This is the appropriate behavior for an implementation which does not support direct access files. See AI-00332.)

  If Yes, then the following tests should report NOT-APPLICABLE:

CE2102H.TST	CE2102K.ADA	CE2104C.ADA
CE2104D.ADA	CE2105B.ADA	CE2106B.ADA
CE2107E.ADA	CE2107F.ADA	CE2107G.ADA
CE2107H.ADA	CE2107I.ADA	CE2108C.ADA
CE2108D.ADA	CE2109B.ADA	CE2111B.ADA
CE2111H.ADA	CE2111E.ADA	CE2111G.ADA
CE2115A.ADA	CE2401A.ADA	CE2401B.ADA
CE2401C.ADA	EE2401D.ADA	CE2401E.ADA
CE2401F.ADA	EE2401G.ADA	CE2401H.ADA
CE2404A.ADA	CE2405B.ADA	CE2406A.ADA
CE2407A.ADA	CE2408A.ADA	CE2409A.ADA
CE2410A.ADA	CE2411A.ADA	

- Does the implementation support dynamic creation and deletion of text files? (Y/N) _____

  If No, then the following tests should report NOT-APPLICABLE:

  CE3110A.ADA
  CE3102B.ADA

- Does the implementation allow more than one internal text file to be associated with the same external file? (Y/N) _____

  If No, then the following tests should report NOT-APPLICABLE:

CE2110B.ADA	CE2111D.ADA	CE3111A.ADA
CE3111B.ADA	CE3111C.ADA	CE3111D.ADA
CE3111E.ADA	CE3114B.ADA	CE3115A.ADA

- Does the implementation allow more than one internal sequential file to be associated with the same external file? (Y/N) _____

  If No, then the following tests should report NOT-APPLICABLE:

CE2107A.ADA	CE2107B.ADA	CE2107C.ADA
CE2107D.ADA	CE2111D.ADA	

  Does the implementation allow more than one internal direct access file to be associated with the same external file? (Y/N) _____

  If No, then the following tests should report NOT-APPLICABLE:

CE2107E.ADA	CE2107F.ADA	CE2107G.ADA
CE2107H.ADA	CE2107I.ADA	CE2111H.ADA

- Does the implementation allow resetting one of several internal sequential files all of which are associated with the same external file? (Y/N) _____

  If No, then the following test should report NOT-APPLICABLE:

  CE3115A.ADA

- Is the INLINE pragma supported for PROCEDURES? (Y/N) _____

  If No, then the following tests should report NOT-APPLICABLE:

  LA3004A*.ADA
  EA3004C*.ADA
  CA3004E*.ADA

- Is the INLINE pragma supported for FUNCTIONS? (Y/N) _____

  If No, then the followi g tests should report NOT-APPLICABLE:

  LA3004B*.ADA
  EA3004D*.ADA
  CA3004F*.ADA

- The following test should report NOT-APPLICABLE if the depth of recursion causes STORAGE_ERROR to be raised:

  D64005F*.ADA

- The following test should report NOT-APPLICABLE if DURATION'FIRST = DURATION'BASE'FIRST and the test otherwise executes properly, or if DURATION'LAST = DURATION'BASE'LAST and the test otherwise executes properly:

  C96005B.TST

- The following test should report NOT-APPLICABLE if it compiles without error and F'MACHINE_OVERFLOWS is false and 2.0 * F'MACHINE_RADIX ** F'MACHINE_EMAX <= F'BASE'LARGE, where F is given by TYPE F IS DIGITS SYSTEM.MAX_DIGITS:

  C4A013B.ADA

- Is the value of SYSTEM.MAX_DIGITS > 35? (Y/N) _____

  If Yes, then the following test is inapplicable:

  C4A011A.ADA

TESTS WHICH MAY NOT REPORT "NOT-APPLICABLE"

- Is mode IN_FILE supported for sequential files? (Y/N) _____

  If Yes, then the following test is inapplicable:

  CE2102D.ADA

- Is mode OUT_FILE supported for sequential files? (Y/N) _____

  If Yes, then the following test is inapplicable:

  CE2102E.ADA

- Is mode INOUT_FILE supported for direct access files? (Y/N) _____

  If Yes, then the following test is inapplicable:

  CE2102F.ADA

- Are RESET and DELETE supported for sequential files? (Y/N) _____

  If Yes, then the following test is inapplicable:

  CE2102G.ADA

- Are RESET and DELETE supported for direct access files? (Y/N) _____

  If Yes, then the following test is inapplicable:

  CE2102K.ADA

- Is mode IN_FILE supported for direct access files? (Y/N) _____

  If Yes, then the following test is inapplicable:

  CE2102I.ADA

- Is mode OUT_FILE supported for direct access files? (Y/N) _____

  If Yes, then the following test is inapplicable:

  CE2102J.ADA

- Are dynamic creation and deletion of sequential files supported? (Y/N) _____

  If Yes, then the following test is inapplicable:

  CE2106A.ADA

- Are dynamic creation and deletion of direct files supported? (Y/N) _____

  If Yes, then the following test is inapplicable:

  CE2106B.ADA

20 July 1987

ACVC VERSION 1.9 WITHDRAWN TESTS

The following tests have been withdrawn from Version 1.9 of the Ada Compiler Validation Capability (ACVC) for the reasons given below:

- C34004A: The expression in line 168 yields a value outside the range of the target type T, but there is no handler for CONSTRAINT_ERROR.

- C35502P: The equality operators in lines 62 and 69 should be inequality operators.

- C37213H: The subtype declaration of SCONS in line 100 is incorrectly expected to raise an exception when elaborated.

- C37213J: The aggregate in line 451 incorrectly raises CONSTRAINT_ERROR.

- C37215C, C37215E, C37215G, and C37215H: Various discriminant constraints are incorrectly expected to be incompatible with type CONS.

- C41402A: The attribute 'STORAGE_SIZE is incorrectly applied to an object of an access type.

- C45614C: The function call of IDENT_INT in line 15 uses an argument of the wrong type.

- BC3105A: Lines 159 through 168 expect error messages, but these lines are correct Ada.

- AD1A01A: The declaration of subtype SINT3 raises CONSTRAINT_ERROR for implementations which select INT'SIZE to be 16 or greater.

- CE2401H: The record aggregates in lines 105 and 117 contain the wrong values.

**** End of List ****
